Math. Program., Ser. A (2019) 175:307–353
https://doi.org/10.1007/s10107-018-1244-x

FULL LENGTH PAPER

Nearly linear-time packing and covering LP solvers

Achieving width-independence and $O(1/\varepsilon)$-convergence

Zeyuan Allen-Zhu[1] · **Lorenzo Orecchia**[2]

Received: 8 August 2016 / Accepted: 30 January 2018 / Published online: 14 February 2018

Abstract Packing and covering linear programs (PC-LPs) form an important class of linear programs (LPs) across computer science, operations research, and optimization. Luby and Nisan (in: STOC, ACM Press, New York, 1993) constructed an iterative algorithm for approximately solving PC-LPs in *nearly linear time*, where the time complexity scales nearly linearly in N, the number of nonzero entries of the matrix, and polynomially in ε, the (multiplicative) approximation error. Unfortunately, existing nearly linear-time algorithms (Plotkin et al. in Math Oper Res 20(2):257–301, 1995; Bartal et al., in: Proceedings 38th annual symposium on foundations of computer science, IEEE Computer Society, 1997; Young, in: 42nd annual IEEE symposium on foundations of computer science (FOCS'01), IEEE Computer Society, 2001; Koufogiannakis and Young in Algorithmica 70:494–506, 2013; Young in Nearly linear-time approximation schemes for mixed packing/covering and facility-location linear programs, 2014. arXiv:1407.3015; Allen-Zhu and Orecchia, in: SODA, 2015) for solving PC-LPs require time at least proportional to ε^{-2}. In this paper, we break this long-standing barrier by designing a packing solver that runs in time $\widetilde{O}(N\varepsilon^{-1})$ and covering LP solver that runs in time $\widetilde{O}(N\varepsilon^{-1.5})$. Our packing solver can be extended to run in

An earlier version of this paper appeared on arXiv http://arxiv.org/abs/1411.1124 in November 2014. A 6-paged abstract of this paper "Nearly-Linear Time Positive LP Solver with Faster Convergence Rate," excluding Sects. 4, 5 and 6, and excluding all the proofs in Sects. 2 and 3, has been presented at the STOC 2015 conference in Portland, OR.

✉ Zeyuan Allen-Zhu
zeyuan@csail.mit.edu

Lorenzo Orecchia
orecchia@bu.edu

1 Microsoft Research AI, Redmond, WA, USA

2 Boston University, Boston, MA, USA

time $\widetilde{O}(N\varepsilon^{-1})$ for a class of well-behaved covering programs. In a follow-up work, Wang et al. (in: ICALP, 2016) showed that all covering LPs can be converted into well-behaved ones by a reduction that blows up the problem size only logarithmically.

Mathematics Subject Classification 90C05, Linear programming · 90C25, Convex programming · 65K05, Mathematical programming methods · 49M20, Methods of relaxation type

1 Introduction

A packing linear program (LP) takes the form $\max\{c^T x \,:\, Ax \leq b\}$ where $c \in \mathbb{R}^n_{\geq 0}$, $b \in \mathbb{R}^m_{\geq 0}$, and $A \in \mathbb{R}^{m\times n}_{\geq 0}$. A covering LP can be written as $\min\{b^T y \,:\, A^T y \geq c\}$, with the same requirements on A, b, and c. We denote by N the number of non-zero elements in matrix A. We assume without loss of generality that the two LP programs are in their *standard forms*:

$$\text{Packing LP:} \quad \max_{x\in\mathbb{R}^n_{\geq 0}} \{\mathbb{1}^T x \,:\, Ax \leq \mathbb{1}\}, \tag{1.1}$$

$$\text{Covering LP:} \quad \min_{y\in\mathbb{R}^m_{\geq 0}} \{\mathbb{1}^T y \,:\, A^T y \geq \mathbb{1}\}. \tag{1.2}$$

The two programs are dual to each other, so we denote by $\mathsf{OPT} \geq 0$ their shared optimum. We say x is a $(1-\varepsilon)$-approximation for the packing LP if $Ax \leq \mathbb{1}$ and $\mathbb{1}^T x \geq (1-\varepsilon)\mathsf{OPT}$, and y a $(1+\varepsilon)$-approximation for the covering LP if $A^T y \geq \mathbb{1}$ and $\mathbb{1}^T y \leq (1+\varepsilon)\mathsf{OPT}$.

In this paper, we study first-order iterative methods for solving packing and covering linear programs (PC-LPs) efficiently.[1] Of course, it is possible to adopt the Interior Point or Ellipsoid methods to obtain approximate solvers with a $\log(1/\varepsilon)$ dependence on the number of iterations. However, the computational cost of such algorithms is typically high, as each iteration requires solving a linear system, and thus is not suitable for large-scale applications.

To address this issue, researchers have developed *iterative approximate* PC-LP solvers that achieve a better dependence on the problem size (e.g., nearly linear in N) at the cost of having a $\mathsf{poly}(1/\varepsilon)$ dependence on the approximation parameter ε. Such iterative solvers have been widely applied in approximation algorithms (e.g., MinSetCover [24], MaxSet, MaxDiCut, Max- k- CSP [32], bipartite matching), probabilistic checkable proofs [32], zero-sum matrix games [29], scheduling [31], graph embedding [31], flow controls [10,11], auction mechanisms [37], wireless sensor networks [14], and many other areas. In addition, techniques developed in this line of research have inspired important results on other fundamental algorithmic problems, such as the design of fast algorithms for multi-commodity flow problems [9,18,19,25,31] and the equivalence between QIP and PSPACE [21].

[1] Luby and Nisan, who originally studied iterative solvers for this class of problems [24], dubbed them *positive LPs*. However, the class of LPs with non-negative constraint matrices is slightly larger, including mixed-packing-and-covering LPs. For this reason, we prefer to stick to the PC-LP terminology.

Table 1 Comparisons among iterative approximate solvers for packing and covering LPs

Paper	Running time	Width independent?	Nearly linear-time?
[31]	$O(N \times \frac{\rho^2 \mathsf{OPT}^2 \log m}{\varepsilon^2})$	No	No
[7]	$O(N \times \frac{\rho \mathsf{OPT} \log m}{\varepsilon^2})$	No	No
[26,29]	$O(N \times \frac{\rho \mathsf{OPT} \log m}{\varepsilon})$	No	No
[13]	$O(N \times \frac{\sqrt{Kn} \log m}{\varepsilon})$	Yes	No
[15,27]: packing LP	$\widetilde{O}(N \times (n + \frac{\sqrt{n}}{\varepsilon}))$	Yes	No
Parallel solvers [4,8,10,11,24,34,35]	$O(N \times \frac{\log^2 N \log(1/\varepsilon)}{\varepsilon^2})$ at best	Yes	Yes
[35]	$O((md + N) \times \frac{\log N}{\varepsilon^2})$	Yes	Almost yes
[10,11]	$O(nm \times \frac{\log N}{\varepsilon^2})$	Yes	Almost yes
[36]	$O(N \times \frac{\log N}{\varepsilon^2})$	Yes	Yes
[23]	$O(N + (n+m) \times \frac{\log N}{\varepsilon^2})$	Yes	Yes
Theorem 3.4 packing LP	$O(N \times \frac{\log N \log \varepsilon^{-1}}{\boldsymbol{\varepsilon}})$	Yes	Yes
Theorem 5.3 well-behaved covering LP	$O(N \times \frac{\log N \log \varepsilon^{-1}}{\boldsymbol{\varepsilon}})$	Yes	Yes
Theorem 6.6 covering LP	$O(N \times \frac{\log N \log \varepsilon^{-1}}{\boldsymbol{\varepsilon}^{1.5}})$	Yes	Yes

The width $\rho \in [1/\mathsf{OPT}, \infty)$ is defined as the largest entry of the constraint matrix A. The parameter d is the maximum number of constraints each variable is in; md may be larger than N

Previous iterative approximate solvers can be divided into two classes, *width-dependent* and *width-independent* solvers (see also Table 1).

Width-dependent solvers [2] Based on multiplicative weight update ideas (a.k.a. exponentiated gradient updates), researchers have obtained solvers for PC-LPs with a running time at least N multiplied with $\rho\mathsf{OPT} \in [1, \infty)$, where ρ is the *width* of the program, i.e., the largest entry of matrix A. For instance, PC-LPs can be solved in $O(\frac{N\rho^2\mathsf{OPT}^2 \log m}{\varepsilon^2})$-time [31], or $O(\frac{N\rho\mathsf{OPT} \log m}{\varepsilon^2})$-time using some more refined analysis [7]. These algorithms only require "oracle-access" to the matrix A. When A is given explicitly like in this paper, the running time can be reduced to $O(\frac{N\rho\mathsf{OPT} \log m}{\varepsilon})$ by deploying Nesterov's accelerated gradient method [29], or Nemirovski's mirror prox method [26].

[2] Most width-dependent solvers study the minmax problem $\min_{x \geq 0, \mathbb{1}^T x = 1} \max_{y \geq 0, \mathbb{1}^T y = 1} y^T A x$, whose optimal value equals $1/\mathsf{OPT}$. Their approximation guarantees are often written in terms of *additive* error. We have translated their performances to multiplicative error for a clear comparison.

Width-dependent algorithms are not polynomial time but only *pseudo-polynomial time*.

Width-independent, but super linear-time solvers Researchers also tried to appropriately scale the matrix so as to avoid the width penalty in the above methods. For instance, Bienstock and Iyengar [13] built on Nesterov's method [29] and obtained a running time $O(\varepsilon^{-1}N\sqrt{Kn\log m})$ where K is the maximum number of non-zeros per row of A. This is $O(\varepsilon^{-1}Nn\sqrt{\log m})$ in the worst case. The results of [15,27] improved this complexity (for packing LP only) to $\widetilde{O}(\varepsilon^{-1}N\sqrt{n})$, at a cost of enduring an $\widetilde{O}(Nn)$-time preprocessing stage.

Width-independent, nearly linear-time solvers Perhaps the most desirable complexity is a running time that is both independent of the width parameter ρ, and also nearly linearly scales with N.[3] This line of research was initiated by a seminal paper of Luby and Nisan [24], who gave an algorithm running in $O\left(\frac{N\log^2 N}{\varepsilon^4}\right)$ time with no dependence on the width ρ. This is also the first *nearly linear-time* approximate solver for PC-LPs, and also the first to run in parallel in nearly linear-work and polylogarithmic depth.

The parallel algorithm of Luby and Nisan was extended by [4,8,10,33,35]. Most notably, the algorithm of Wang *et al.* [33] runs in $O(\frac{\log^2 N\log(1/\varepsilon)}{\varepsilon^2})$ iterations, each costing a matrix-vector multiplication that can be implemented in $O(N)$ total work.

The ideas of Luby and Nisan also led to sequential width-independent, nearly linear-time PC-LP solvers [10,11,23,35,36]. Most notably, the algorithm of Koufogiannakis and Young [23] runs in time $O\left(N+\frac{\log N}{\varepsilon^2}\times(n+m)\right)$.

Despite the amount of work in this area, the $O(1/\varepsilon^2)$ convergence rate was established in 1997 [10,11] and has not been improved since then. On a separate note, Klein and Young [22] showed that all Dantzig-Wolfe type algorithms have to suffer from a $O(1/\varepsilon^2)$ convergence rate. This lack of progress constitutes a significant limitation, as the ε^{-2}-dependence (also known as the $1/\sqrt{T}$ convergence) on the approximation parameter ε is particularly poor.

1.1 Our results

Packing LP We present an algorithm *PacLPSolver* that runs in $O(\frac{\log(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon}N)$ total time. This gives the first width-independent, and the first nearly linear-time solver for packing LP with an ε^{-1} convergence (i.e., an $1/T$ convergence). In contrast, no nearly linear-time algorithm has achieved any convergence rate faster than ε^{-2} before our work.

Interestingly, the maximum (weighted) bipartite matching is just one instance of a packing LP. As a consequence, our *PacLPSolver* algorithm finds an approximate maximum bipartite matching in time $\widetilde{O}(m\varepsilon^{-1})$. This new matching algorithm, which arises purely from convex-optimization arguments, matches the running time of the best known combinatorial algorithm for maximum weighted bipartite matching [16].

[3] Some of these solvers still have a polylog(ρ) dependence. Since each occurrence of $\log(\rho)$ can be replaced with $\log(nm)$ after slightly modifying the matrix A, we have done so in Table 1 for a fair comparisons.

Covering LP A symmetric design of *PacLPSolver* gives rise to an algorithm $CovLPSolver^{\mathsf{wb}}$ with the same running time $O(\frac{\log(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon}N)$, but only solving *well-behaved* covering LP instances. At a high level, we say an instance is well-behaved if the constraint $A^T y \geq \mathbb{1}$ is "never redundant": for instance, if the optimal solution y^* satisfies $C \cdot \mathbb{1} \geq A^T y^* \geq \mathbb{1}$ for some constant $C > 1$ then the covering LP is well-behaved. For the *general* covering LP *without* well-behavior assumptions, we propose a different algorithm *CovLPSolver* that runs in time $O(\frac{\log(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon^{1.5}}N)$. Again, we emphasize that no nearly linear-time covering LP solver can achieve a convergence rate faster than ε^{-2} (or equivalently $O(1/\sqrt{T})$) before our work.

REMARK After the first version of this paper appeared on arXiv in 2014, Wang, Rao and Mahoney [34] showed all covering LPs can be converted into well-behaved ones, by blowing up the problem size logarithmically. In other words, they obtained a nearly linear-time covering LP solver with ε^{-1} convergence by a reduction to $CovLPSolver^{\mathsf{wb}}$. Nevertheless, our *CovLPSolver*, being a direct method, may still be of practical and theoretical interests.

1.2 Main challenge and our approach

Width-independence versus acceleration Previous solvers for PC-LPs are based on standard techniques in non-smooth optimization. They first implicitly or explicitly smoothen the objective, often by the entropy regularizer. Then, they minimize the resulting convex objective either via variations of full-gradient methods, yielding parallel algorithms, or via variations of coordinate-gradient methods, yielding sequential algorithms. The *main challenge* in previous work is to show that the width dependence can sometimes be completely removed for PC-LPs, if the underlying minimization method is designed cleverly.

Of course, the slower the convergence rate is, the easier it is to design nearly linear-time solvers. The ε^{-4}-convergence solver of Awerbuch and Khandekar [8] and the ε^{-3}-convergence solver of [4] are arguably the simplest nearly linear-time solvers at this point.

In this paper, we achieve the ε^{-1} convergence that is typical for accelerated gradient descent over smoothened objectives [29], but without paying the width or any additional super-logarithmic factors. The challenge in this approach is to preserve the width-independence and the accelerated rate *at the same time*. We stress here that our algorithm is *not an instance* of any known variant of accelerated gradient descent.[4] Moreover, the incorporation of width-independence and Nesterov's acceleration requires significant effort, as witnessed by the lack of progress on this problem for the last 15 years.

Our high-level approach Our approach is based on an improved convex formalization $f(x)$ of the PC-LP objective, together with our linear-coupling framework for designing efficient first-order methods [5] for minimizing $f(x)$.

[4] This can be verified by observing that our objective $f_\mu(x)$, to be introduced later, is not globally Lipschitz smooth, so that one cannot apply accelerated gradient descent directly.

The improved formalization shows that our smoothened objective $f(x)$ satisfies either the classical condition for Lipschitz smoothness or a different condition based on multiplicative change. This formalization also clarifies why width-independent algorithms exist in the first place. See Lemma 2.7 and the related discussion for more details.

The linear-coupling framework in our previous work [5] provides a different interpretation of Nesterov's acceleration for smooth optimization [28]. In a nutshell, this linear-coupling framework allows us to construct accelerated algorithms by coupling the executions of a gradient descent algorithm, yielding iterates $\{\mathsf{y}_k\}$ and a mirror descent step algorithm, with iterates $\{\mathsf{z}_k\}$. The name "linear coupling" stems from the fact that, at iteration $k+1$, the gradient of the objective is queried at a point x_{k+1}, which is a linear combination of gradient and mirror steps, i.e., $\mathsf{x}_{k+1} = (1-\tau)\cdot z_k + (1-\tau)\cdot y_k$.

In this paper, we apply linear coupling in a very non-trivial manner. We design a gradient and a mirror descent step, each very specific to the underlying PC-LP problem. We also perform a coupling step $\mathsf{x}_{k+1} = (1-\tau)\cdot z_k + (1-\tau)\cdot y_k$, but need to design a different analysis to preserve width independence. None of these components has appeared in [5].

Arithmetic precision Throughout this paper, we assume exact arithmetic operations for presenting the cleanest proofs. If the updates are calculated within precision $\frac{1}{\mathsf{poly}(\varepsilon^{-1},n,m)}$, or equivalently when word size $O(\log(\varepsilon^{-1}+n+m))$ is used, our results still hold.[5]

Roadmap We relax the packing LP in Sect. 2, and provide our packing LP solver in Sect. 3. We relax the covering LP in Sect. 4, and provide our covering LP solver in the well-behaved case in Sect. 5. In Sect. 6, we provide our full covering LP solver.

2 Relaxation of the packing linear program

To solve packing LP, we minimize a relaxed version of the original LP, where the hard constraint $Ax \le 1$ is regularized by entropy and replaced by an exponential penalty function.

Notations Recall that the packing LP in its standard form is $\max_{x\ge 0}\{\mathbb{1}^T x : Ax \le \mathbb{1}\}$. Let us denote by OPT the optimal value of this linear program, and x^* any optimal solution. We say that x is a $(1-\varepsilon)$-approximation for the packing LP if $Ax \le \mathbb{1}$ and $\mathbb{1}^T x \ge (1-\varepsilon)\mathsf{OPT}$.

Throughout this paper, we use the indices $i \in [n]$ to denote the columns of A, and the indices $j \in [m]$ to denote the rows of A. We let $A_{:i}$ be the i-th column

[5] Due to space limitation, we quickly sketch why logarithmic word size suffices for our algorithms. On one hand, one can prove in an iteration, if x is calculated with a small additive error $1/\mathsf{poly}(1/\varepsilon, n, m)$, then the objective $f(x)$ may increase only by $1/\mathsf{poly}(1/\varepsilon, n, m)$ in that iteration. The proof of this relies on the fact that (1) one can assume without loss of generality all entries of A are no more than $\mathsf{poly}(1/\varepsilon, n, m)$ and (2) our algorithms ensure $f(x) < \mathsf{poly}(1/\varepsilon, n, m)$ for all iterations with high probability, so even though we are using the exponential functions, $f(x)$ will not change additively by much. On the other hand, one can similarly prove that each $\nabla_i f(x)$ can be calculated within an additive error $1/\mathsf{poly}(1/\varepsilon, n, m)$ in each iteration. They together imply that the total error incurred by arithmetic operations can be made negligible.

vector of A, and $A_{j:}$ the j-th row vector of A. Given any vector x, we denote by $\|x\|_A = \sqrt{\sum_{i\in[n]} x_i^2 \cdot \|A_{:i}\|_\infty}$ the A-norm of x. By simple scaling, we can assume without loss of generality that [6]

$$\min_{i\in[n]}\{\|A_{:i}\|_\infty\} = 1. \tag{2.1}$$

We restrict the domain of x and the range of OPT as follows.

Fact 2.1 *Define the bounding box* $\Delta_{\mathsf{box}} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : x_i \in \left[0, \frac{1}{\|A_{:i}\|_\infty}\right]\}$. *Under assumption* (2.1)*, we have* $\mathsf{OPT} \in [1, n]$ *and* $\{x : x \geq 0 \wedge Ax \leq \mathbb{1}\} \subseteq \Delta_{\mathsf{box}}$.

Proof Suppose that i^* is the column that achieves the smallest infinite norm $\|A_{:i}\|_\infty$ over all columns. Letting x be such that $x_i = 1$ at $i = i^*$ and $x_i = 0$ at $i \neq i^*$, we claim that x is a feasible solution for the packing LP (1.1), simply because $\|A_{:i^*}\|_\infty = 1$ according to (2.1). This feasible solution x yields an objective value $\mathbb{1}^T x = 1$, proving that $\mathsf{OPT} \geq 1$. On the other hand, for any solution $x \geq 0$ satisfying $Ax \leq \mathbb{1}$, we must have $x_i \leq \frac{1}{\|A_{:i}\|_\infty}$ for each i. Therefore, $\mathbb{1}^T x \leq \sum_i \frac{1}{\|A_{:i}\|_\infty} \leq n$, proving that $\mathsf{OPT} \leq n$.

The inclusion $\{x : x \geq 0 \wedge Ax \leq \mathbb{1}\} \subseteq \Delta_{\mathsf{box}}$ is obvious, since the constraints $x \geq 0$ and $Ax \leq \mathbb{1}$ together imply $x_i \leq \frac{1}{\|A_{:i}\|_\infty}$ for every $i \in [n]$. □

This bounding-box constraint allows us to focus only on searching x in Δ_{box}.

Our regularized objective We now introduce the smoothed objective $f_\mu(x)$ that we minimize over Δ_{box} in order to approximately solve packing LP. At a high level, this objective $f_\mu(x)$ turns each row of the hard, non-smooth LP constraint $Ax \leq \mathbb{1}$ into an exponential penalty function so that we only need to require $x \in \Delta_{\mathsf{box}}$ throughout the algorithm.

Formally, the packing LP can be written as the following minimization problem by introducing the Lagrangian variable $y \in \mathbb{R}^m$:

$$\min_{x\in\Delta_{\mathsf{box}}} \left\{ -\mathbb{1}^T x + \max_{y\geq 0}\{y^T Ax - \mathbb{1}^T y\}\right\}. \tag{2.2}$$

The problem can be now smoothened by introducing a concave regularizer over $y \geq 0$. We take this regularizer to be the generalized entropy $H(y) = -\sum_{j=1}^m y_j \log y_j + y_j$ over the first orthant $y \geq 0$, and minimize the following smoothened objective $f_\mu(x)$ over $x \in \Delta_{\mathsf{box}}$:

$$f_\mu(x) \stackrel{\text{def}}{=} -\mathbb{1}^T x + \max_{y\geq 0}\{y^T Ax - \mathbb{1}^T y + \boxed{\mu \cdot H(y)}\}. \tag{2.3}$$

Above, $\mu > 0$ is some smoothing parameter to be chosen later. By explicitly computing the maximization over $y \geq 0$, $f_\mu(x)$ can be rewritten as

[6] If $\min_{i\in[n]}\{\|A_{:i}\|_\infty\} = 0$ then the packing LP is unbounded so we are done. Otherwise, if $\min_{i\in[n]}\{\|A_{:i}\|_\infty\} = v > 0$ we scale all entries of A by $1/v$, and scale OPT by v.

Fact 2.2 $f_\mu(x) = \mu \sum_{j=1}^m e^{\frac{1}{\mu}((Ax)_j - 1)} - \mathbb{1}^T x$.

We study the *minimization* problem on $f_\mu(x)$ over $x \in \Delta_{\mathsf{box}}$. Intuitively $f_\mu(x)$ captures the original packing LP (1.1) as follows. Firstly, since we want to maximize $\mathbb{1}^T x$, the negative term $-\mathbb{1}^T x$ shows up in $f_\mu(x)$. Secondly, if a packing constraint $j \in [m]$ is violated by ε, that is, $(Ax)_j \geq 1 + \varepsilon$, the exponential penalty in $f_\mu(x)$ introduces a penalty at least $\mu e^{\varepsilon/\mu}$; this will be a large penalty if $\mu \leq O(\varepsilon / \log(n/\varepsilon))$.

Remark 2.3 The use of exponential function at least traces back to [31] in 1991 (implicitly) and to [20] in 1994 (explicitly). The way most previous results minimize $f_\mu(x)$ is by taking a logarithm $g(x) = \log\left(\sum_{j=1}^m e^{((Ax)_j - 1)/\mu}\right)$, explicitly or implicitly arguing that $g(x)$ is Lipschitz smooth (i.e., $\|\nabla^2 f(x)\|$ is bounded), and then taking gradient descent. [7] Unfortunately, the Lipschitz smoothness parameter of $g(x)$ depends on the width of the LP, and thus first-order iterative approaches based on directly minimizing $g(x)$ are mostly width-dependent [7,26,29,31]. One can also reduce the width parameter of $g(x)$ which yields super linear-time solvers [13,15,27].

In this paper, we directly perform gradient descent and mirror descent on $f_\mu(x)$—without taking the logarithm. Note that traditional accelerated gradient methods [28,29] should not be applied directly to minimize f_μ because it is not Lipschitz smooth. [8]

Our $f_\mu(x)$ incurs a regularization error. The next proposition bounds this error following a similar treatment in [4].

Proposition 2.4 *Let* $\mu = \frac{\varepsilon}{4\log(nm/\varepsilon)}$ *and recall* x^* *is an optimal solution for packing LP.*

(a) $f_\mu(u^*) \leq -(1-\varepsilon)\mathsf{OPT}$ *for* $u^* \stackrel{\text{def}}{=} (1-\varepsilon/2)x^* \in \Delta_{\mathsf{box}}$.
(b) $f_\mu(x) \geq -(1+\varepsilon)\mathsf{OPT}$ *for every* $x \in \Delta_{\mathsf{box}}$.
(c) *If* $x \in \Delta_{\mathsf{box}}$ *satisfies* $f_\mu(x) \leq -(1-\theta)\mathsf{OPT}$ *for some* $\theta \in [0,1]$, *then* $\frac{1}{1+\varepsilon}x$ *is a* $\frac{1-\theta}{1+\varepsilon}$*-approximate solution to the packing LP.*

Remark 2.5 Our box constraint $x \in \Delta_{\mathsf{box}}$ is almost redundant for minimizing $f_\mu(x)$: whenever $x \geq 0$ and $f_\mu(x) \leq 0$, one should automatically have $x_i \leq \frac{1+\varepsilon}{\|A_{:i}\|_\infty}$. However, this constraint shall be used to make sure that our updates are always inside Δ_{box}.

Proof of Proposition 2.4 (a) We have $\mathbb{1}^T u^* = (1-\varepsilon/2)\mathsf{OPT}$ by the definition of OPT. from the feasibility $Ax^* \leq \mathbb{1}$ in the packing LP, we have $Au^* - \mathbb{1} \leq -\varepsilon/2 \cdot \mathbb{1}$, and can compute $f_\mu(u^*)$ as follows:

[7] Note that some of the previous results (such as [7,31]) appear to directly minimize $\sum_{j=1}^m e^{((Ax)_j - 1)/\mu}$ as opposed to its logarithm $g(x)$. However, their per-iteration objective decrease is multiplicative, meaning it is essentially equivalent to performing a single gradient-descent step on $g(x)$ with additive objective decrease.

[8] The exact same $f_\mu(x)$ also appeared in our previous work [4], albeit without this smoothing interpretation and without the constraint $x \in \Delta_{\mathsf{box}}$. The techniques in [4] only leads to ε^{-2} convergence (see Table 1).

$$f_\mu(u^*) = \mu \sum_j e^{\frac{1}{\mu}((Au^*)_j - 1)} - \mathbb{1}^T u^*$$
$$\leq \mu \sum_j e^{\frac{-\varepsilon/2}{\mu}} - (1 - \varepsilon/2)\mathsf{OPT}$$
$$\leq \frac{\mu m}{(nm)^2} - (1 - \varepsilon/2)\mathsf{OPT} \leq -(1 - \varepsilon)\mathsf{OPT}.$$

(b) Suppose towards contradiction that $f_\mu(x) < -(1+\varepsilon)\mathsf{OPT}$. Since $f_\mu(x) > -\mathbb{1}^T x$, it must satisfy that $\mathbb{1}^T x > (1+\varepsilon)\mathsf{OPT}$. Suppose that $\mathbb{1}^T x = (1+v)\mathsf{OPT}$ for some $v > \varepsilon$. By the definition of OPT, we must have that $Ax < (1+v)\mathbb{1}$ is broken, and therefore there exists some $j \in [m]$ satisfying that $(Ax)_j \geq 1 + v$. In such a case, the objective

$$f_\mu(x) \geq \mu e^{v/\mu} - (1+v)\mathsf{OPT} = \frac{\varepsilon}{4\log(nm/\varepsilon)} \left(\left(\frac{nm}{\varepsilon} \right)^4 \right)^{v/\varepsilon} - (1+v)\mathsf{OPT}$$
$$\geq \left(\left(\left(\frac{nm}{\varepsilon} \right)^2 \right)^{v/\varepsilon} - (1+v) \right) \mathsf{OPT} > 0$$

giving a contradiction to the assumption that $f_\mu(x) < 0$.

(c) Note that x satisfies $f_\mu(x) \leq -(1-\delta)\mathsf{OPT} \leq 0$, and we first show $Ax \leq (1+\varepsilon)\mathbb{1}$. Let us assume that $v = \max_j((Ax)_j - 1) \geq 0$ because otherwise we will have $Ax \leq \mathbb{1}$. Under this definition, we have $Ax \leq (1+v)\mathbb{1}$ and therefore $\mathbb{1}^T x \leq (1+v)\mathsf{OPT}$ by the definition of OPT. We compute $f_\mu(x)$ as follows.

$$f_\mu(x) \geq \mu e^{\frac{v}{\mu}} - (1+v)\mathsf{OPT} \geq \mu \left(\left(\frac{nm}{\varepsilon} \right)^4 \right)^{v/\varepsilon} - (1+v)n$$
$$= \frac{\varepsilon}{4\log(nm/\varepsilon)} \left(\left(\frac{nm}{\varepsilon} \right)^4 \right)^{v/\varepsilon} - (1+v)n.$$

The above quantity is positive whenever $v \geq \varepsilon$, and therefore, to satisfy $f_\mu(x) \leq 0$ we must have $v \leq \varepsilon$, which is equivalent to $Ax \leq (1+\varepsilon)\mathbb{1}$. Next, because $-\mathbb{1}^T x \leq f_\mu(x) \leq -(1-\delta)\mathsf{OPT}$, we know $\mathbb{1}^T x \geq (1-\delta)\mathsf{OPT}$. Letting $x' = \frac{1}{1+\varepsilon}x$, we both have that x' is feasible (i.e., $Ax' \leq \mathbb{1}$), and x' has an objective $\mathbb{1}^T x'$ at least as large as $\frac{1-\delta}{1+\varepsilon}\mathsf{OPT}$.

□

Some non-standard smoothness properties The gradient and Hessian of $f_\mu(x)$ can be written in the following closed forms:

Fact 2.6 *$\nabla f_\mu(x) = A^T p(x) - \mathbb{1}$ and $\nabla^2 f_\mu(x) = \frac{1}{\mu} A^T \mathrm{diag}\{p(x)\} A$, where $p_j(x) \stackrel{\text{def}}{=} e^{\frac{1}{\mu}((Ax)_j - 1)}$.*

By staring at these closed forms, we note that $f_\mu(x)$ is not Lipschitz-smooth: for instance, each $\nabla^2_{ii} f_\mu(x)$ can go to infinity so the spectral norm of $\nabla^2 f_\mu(x)$ is

unbounded. However, the non-negativity of A guarantees that whenever $\nabla^2_{ii} f_\mu(x)$ is large for some coordinate i, the corresponding entry of the gradient $\nabla_i f_\mu(x)$ must also be large. This still allows us to take a larger step in direction $\mathbf{e}_i$ than traditionally allowed by coordinate descent.

The above intuition is formalized in the next lemma, whose proof is by simple manipulation of Hessian. The first half of the lemma is the same as the traditional coordinate Lipschitz-smoothness property, but holds only conditionally; the second half is a salient characteristic of this work and requires the non-negativity of A. These smoothness properties will be crucial in applying gradient descent arguments in Sect. 3.3, and are the main motivation for us to adopt the $\|\cdot\|_A$ norm for our proposed algorithms.

Lemma 2.7 *Let* $L \stackrel{\text{def}}{=} \frac{4}{\mu}$. *Then, for every* $x \geq 0$, *every* $i \in [n]$, *and every* $\lambda \in \left[-\frac{1}{L\|A_{:i}\|_\infty}, \frac{1}{L\|A_{:i}\|_\infty}\right]$:

(a) *If* $|\nabla_i f_\mu(x)| \leq 1$, *then* $\left|\nabla_i f_\mu(x+\lambda\mathbf{e}_i) - \nabla_i f_\mu(x)\right| \leq L\|A_{:i}\|_\infty \cdot |\lambda|$.

(b) *If* $\nabla_i f_\mu(x) \geq 1$, *then* $\nabla_i f_\mu(x+\lambda\mathbf{e}_i) \geq \left(1 - \frac{\|A_{:i}\|_\infty L}{2}|\lambda|\right)\nabla_i f_\mu(x)$.

Proof of Lemma 2.7 Using the fact that $\nabla_i f_\mu(x) > -1$ for all x, we have:

$$\begin{aligned}\left|\log\frac{\nabla_i f_\mu(x+\lambda\mathbf{e}_i)+1}{\nabla_i f_\mu(x)+1}\right| &\overset{①}{=} \left|\int_0^\lambda \frac{\nabla^2_{ii} f_\mu(x+\nu\mathbf{e}_i)}{\nabla_i f_\mu(x+\nu\mathbf{e}_i)+1}d\nu\right| \\ &\overset{②}{=} \frac{1}{\mu}\left|\int_0^\lambda \frac{(A^T\mathrm{diag}\{p(x+\nu\mathbf{e}_i)\}A)_{ii}}{(A^T p(x+\nu\mathbf{e}_i))_i}d\nu\right| \\ &\overset{③}{\leq} \frac{\|A_{:i}\|_\infty}{\mu}|\lambda| \overset{④}{=} \frac{\|A_{:i}\|_\infty L}{4}|\lambda|.\end{aligned}$$

Above, ① holds because $\int_0^\lambda g'(\nu)d\nu = g(\lambda) - g(0)$ where $g(\nu) = \log(\nabla_i f_\mu(x+\nu\mathbf{e}_i)+1)$; ② holds according to Fact 2.6; ③ is because the numerator is $\sum_j A^2_{j,i} p_j$ while the denominator is $\sum_j A_{j,i} p_j$; ④ holds because $L = \frac{4}{\mu}$. This immediately implies

$$e^{-\frac{\|A_{:i}\|_\infty L}{4}|\lambda|} \leq \frac{\nabla_i f_\mu(x+\lambda\mathbf{e}_i)+1}{\nabla_i f_\mu(x)+1} \leq e^{\frac{\|A_{:i}\|_\infty L}{4}|\lambda|}.$$

Our assumption on λ implies $\frac{\|A_{:i}\|_\infty L}{4}|\lambda| \leq \frac{1}{4}$, so that we can use the approximation $x \leq e^x - 1 \leq 1.2x$ over $x \in [-\frac{1}{4}, \frac{1}{4}]$. This yields the simpler bound:

$$-\frac{\|A_{:i}\|_\infty L}{4}|\lambda| \leq \frac{\nabla_i f_\mu(x+\lambda\mathbf{e}_i) - \nabla_i f_\mu(x)}{\nabla_i f_\mu(x)+1} \leq 1.2\frac{\|A_{:i}\|_\infty L}{4}|\lambda|.$$

(a) Assuming that $\nabla_i f_\mu(x) \in (-1, 1]$, we have:

$$\left|\nabla_i f_\mu(x+\lambda\mathbf{e}_i) - \nabla_i f_\mu(x)\right| \leq 2.4 \cdot \frac{\|A_{:i}\|_\infty L}{4}|\lambda| \leq \|A_{:i}\|_\infty L|\lambda|.$$

(b) Assuming $\nabla_i f_\mu(x) \geq 1$, we have

$$\begin{aligned}\nabla_i f_\mu(x + \lambda \mathbf{e}_i) &\geq \nabla_i f_\mu(x) - \frac{\|A_{:i}\|_\infty L}{4}|\lambda| \left(\nabla_i f_\mu(x) + 1\right) \\ &\geq \left(1 - \frac{\|A_{:i}\|_\infty L}{2}|\lambda|\right) \nabla_i f_\mu(x).\end{aligned}$$

□

Initialization Iterative methods require a starting point, and we use the following one

Fact 2.8 *Let* $x_i^{\mathsf{start}} \stackrel{\text{def}}{=} \frac{1-\varepsilon/2}{n\|A_{:i}\|_\infty}$ *for each* $i \in [n]$. *Then,* $x^{\mathsf{start}} \in \Delta_{\mathsf{box}}$ *and* $f_\mu(x^{\mathsf{start}}) \leq -\frac{1-\varepsilon}{n}$.

Proof Using the fact that $Ax^{\mathsf{start}} - \mathbb{1} \leq -\varepsilon/2 \cdot \mathbb{1}$, we compute $f_\mu(x^{\mathsf{start}})$ as follows:

$$\begin{aligned}f_\mu(x^{\mathsf{start}}) &= \mu \sum_j e^{\frac{1}{\mu}((Ax^{\mathsf{start}})_j - 1)} - \mathbb{1}^T x^{\mathsf{start}} \leq \mu \sum_j e^{\frac{-\varepsilon/2}{\mu}} - \frac{1-\varepsilon/2}{n} \\ &\leq \frac{\mu m}{(nm)^2} - \frac{1-\varepsilon/2}{n} \leq -\frac{1-\varepsilon}{n}.\end{aligned}$$

Above, we have used $\mathbb{1}^T x^{\mathsf{start}} \geq x_i^{\mathsf{start}} = \frac{1-\varepsilon/2}{n}$, where i is the column s.t. $\|A_{:i}\|_\infty = 1$. □

3 Our packing LP solver

Recall traditional (accelerated or not) gradient descent [28,29] or coordinate descent [6,17,30] should not be applied directly to minimize f_μ, because f_μ is not Lipschitz-smooth.

Our proposed algorithm *PacLPSolver* starts with some initial vector $\mathsf{x}_0 = \mathsf{y}_0 = x^{\mathsf{start}}$ (see Fact 2.8) and $\mathsf{z}_0 = 0$, and is divided into T iterations. In each iteration k, it computes a weighted midpoint $\mathsf{x}_k \leftarrow \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$ for some parameter $\tau \in (0, 1)$. This step is analogous to that in traditional accelerated coordinate descent [6, 17,30]. We then compute y_k and z_k as follows.

We select $i \in [n]$ uniformly at random. Let $\xi_k^{(i)} = (0, \ldots, 0, \mathbb{T}^{\mathsf{p}}(v), 0, \ldots, 0)$ be the vector that is only non-zero at coordinate i, where $v = \nabla_i f_\mu(\mathsf{x}_k) \in [-1, \infty)$, and $\mathbb{T}^{\mathsf{p}}(v)$ is the thresholding function $\mathbb{T}^{\mathsf{p}}(v) \stackrel{\text{def}}{=} \min\{v, 1\}$. We refer to $\xi_k^{(i)}$ as the *truncated gradient*. [9] Next,

- Perform a *mirror (descent) step* $\mathsf{z}_k \leftarrow \mathsf{z}_k^{(i)} \stackrel{\text{def}}{=} \operatorname{argmin}_{z \in \Delta_{\mathsf{box}}} \left\{\frac{1}{2}\|z - \mathsf{z}_{k-1}\|_A^2 + \langle n\alpha_k \xi_k^{(i)}, z\rangle\right\}$ for some parameter $\alpha_k \ll 1/n$ to be chosen later.

[9] A similar gradient truncation was developed in our prior work [4], but for a different purpose (to ensure parallelism) and not applied to coordinate gradient. The truncation idea of this paper also inspired later works in matrix scaling [2] and in SDP [1].

Algorithm 1 *PacLPSolver*$(A, x^{\mathsf{start}}, \varepsilon)$

Input: $A \in \mathbb{R}^{m\times n}_{\geq 0}$, $x^{\mathsf{start}} \in \Delta_{\mathsf{box}}$, $\varepsilon \in (0, 1/30]$.

Output: $x \in \Delta_{\mathsf{box}}$. ▷ recall $\Delta_{\mathsf{box}} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : x_i \in [0, \frac{1}{\|A_{:i}\|_\infty}]\}$

1: $\mu \leftarrow \frac{\varepsilon}{4\log(nm/\varepsilon)}$, $L \leftarrow \frac{4}{\mu}$, $\tau \leftarrow \frac{1}{3\cdot nL}$ and $\alpha_0 \leftarrow \frac{1}{nL}$. ▷ parameters

2: $T \leftarrow \lceil 3nL\log(1/\varepsilon)\rceil = O(n\cdot\frac{\log(nm/\varepsilon)\cdot\log(1/\varepsilon)}{\varepsilon})$. ▷ number of iterations

3: $\mathsf{x}_0 = \mathsf{y}_0 \leftarrow x^{\mathsf{start}}$, $\mathsf{z}_0 \leftarrow 0$.

4: **for** $k \leftarrow 1$ **to** T **do**

5: $\quad \alpha_k \leftarrow \frac{1}{1-\tau}\alpha_{k-1}$

6: $\quad \mathsf{x}_k \leftarrow \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$.

7: $\quad$ Randomly select $i \in [n]$ uniformly at random.

8: $\quad$ Define vector $\xi_k^{(i)}$ to be all-zero except at coordinate i, where $\xi_{k,i}^{(i)} = \min\{1, \nabla_i f_\mu(\mathsf{x}_k)\}$.

9: $\quad \mathsf{z}_k \leftarrow \mathsf{z}_k^{(i)} \stackrel{\text{def}}{=} \operatorname{argmin}_{z\in\Delta_{\mathsf{box}}}\{\frac{1}{2}\|z-\mathsf{z}_{k-1}\|_A^2 + \langle n\alpha_k\xi_k^{(i)}, z\rangle\}$. ▷ See Proposition 3.2

10: $\quad \mathsf{y}_k \leftarrow \mathsf{y}_k^{(i)} \stackrel{\text{def}}{=} \mathsf{x}_k + \frac{1}{n\alpha_k L}(\mathsf{z}_k^{(i)} - \mathsf{z}_{k-1})$.

11: **end for**

12: **return** y_T.

– Perform a *gradient (descent) step* $\mathsf{y}_k \leftarrow \mathsf{y}_k^{(i)} \stackrel{\text{def}}{=} \mathsf{x}_k + \frac{1}{n\alpha_k L}(\mathsf{z}_k^{(i)} - \mathsf{z}_{k-1})$.

This finishes the description of our *PacLPSolver*.

Remark 3.1 We use the superscript $^{(i)}$ on $\xi_k^{(i)}$, $\mathsf{y}_k^{(i)}$ and $\mathsf{z}_k^{(i)}$ to emphasize that the value depends on the choice of i. We use generic parameters τ, α_k, T in the above description and their precise values are presented in Algorithm 1.

Our update on y_k is a "gradient descent step" because we shall prove that it strictly decreases the objective (i.e., $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \geq 0$). Our update on z_k is a "mirror descent step" because we shall apply standard mirror descent analysis [12] to it. We explicitly describe how to implement the mirror step (its proof is straightforward by computing the gradient):

Proposition 3.2 *If* $\Delta_{\mathsf{box}} = \{x \in \mathbb{R}^n : x_i \in [0, \frac{C}{\|A_{:i}\|_\infty}]\}$ *for some constant* $C > 0$, *the minimizer* $z = \operatorname{argmin}_{z\in\Delta_{\mathsf{box}}}\{\frac{1}{2}\|z-\mathsf{z}_{k-1}\|_A^2 + \langle\delta\mathbf{e}_i, z\rangle\}$ *for any* $\delta \in \mathbb{R}$ *and basis vector* $\mathbf{e}_i$ *can be computed as follows:*

1. $z \leftarrow \mathsf{z}_{k-1}$.
2. $z_i \leftarrow z_i - \delta/\|A_{:i}\|_\infty$.
3. *If* $z_i < 0$, *then* $z_i \leftarrow 0$; *if* $z_i > C/\|A_{:i}\|_\infty$, $z_i \leftarrow C/\|A_{:i}\|_\infty$.
4. *Return* z.

We also point out that

Lemma 3.3 *Each iteration of PacLPSolver can be implemented to run in expected* $O(N/n)$ *time. The total expected running time is* $O(TN/n)$.

Lemma 3.3 is not hard to prove, but anyways included in "Appendix E.5". It follows from standard implementation tricks which compute x_k and y_k only *implicitly*: that is to express x_k and y_k as linear combinations of two less-frequently-updated vectors.

3.1 Convergence statement

In this section, we focus on proving the following main theorem.

Theorem 3.4 *PacLPSolver*$(A, x^{\mathsf{start}}, \varepsilon)$ *outputs some* y_T *satisfying*

$$\mathbf{E}[f_\mu(\mathsf{y}_T)] \le -(1-3\varepsilon)\mathsf{OPT}.$$

It is straightforward to use Markov's bound to turn Theorem 3.4 into a probabilistic one

Corollary 3.5 *With probability at least* $2/3$*, the output* $\mathsf{y}_T = PacLPSolver(A, x^{\mathsf{start}}, \varepsilon)$ *satisfies that* $\frac{\mathsf{y}_T}{1+\varepsilon}$ *is a* $(1-O(\varepsilon))$ *approximate solution to the packing LP program. The expected running time is* $O(\frac{\log(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon}N)$.

Proof of Corollary 3.5 Since for every $x \in \Delta_{\mathsf{box}}$ it satisfies $f_\mu(x) \ge -(1+\varepsilon)\mathsf{OPT}$ according to Proposition 2.4.b, we obtain that $f_\mu(y_T) + (1+\varepsilon)\mathsf{OPT}$ is a random variable that is non-negative, whose expectation $\mathbf{E}[f_\mu(y_T) + (1+\varepsilon)\mathsf{OPT}] \le 4\varepsilon$ according to Theorem 3.4. By Markov bound, with at least probability $2/3$, we obtain some y_T satisfying $f_\mu(y_T) \le -(1-11\varepsilon)\mathsf{OPT}$, which yields a $(1-O(\varepsilon))$ approximate solution to packing LP according to Proposition 2.4.c. The running time follows from Lemma 3.3. □

Before we prove prove Theorem 3.4 in subsequent subsections, let us first point out that our iterates $\mathsf{x}_k, \mathsf{y}_k, z_k$ never leave the bounding box Δ_{box}:

Lemma 3.6 *We have* $\mathsf{x}_k, \mathsf{y}_k, \mathsf{z}_k \in \Delta_{\mathsf{box}}$ *for all* $k = 0, 1, \ldots, T$.

(The proof of Lemma 3.6 is included in "Appendix A.1", and the main technique already appeared in randomized coordinate descent [17].)

3.2 Step 1: mirror descent guarantee

Following almost classical analysis of mirror descent (cf. textbook [12]), our update $\mathsf{z}_k^{(i)} = \mathrm{argmin}_{z\in\Delta_{\mathsf{box}}}\left\{\frac{1}{2}\|z-\mathsf{z}_{k-1}\|_A^2 + \langle n\alpha_k\xi_k^{(i)}, z\rangle\right\}$ satisfies

Lemma 3.7 *(mirror descent) For every* $u \in \Delta_{\mathsf{box}}$*, it satisfies*

$$\left\langle n\alpha_k\xi_k^{(i)}, \mathsf{z}_{k-1}-u\right\rangle \le n^2\alpha_k^2 L\cdot\left\langle\xi_k^{(i)}, \mathsf{x}_k-\mathsf{y}_k^{(i)}\right\rangle + \frac{1}{2}\|\mathsf{z}_{k-1}-u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k^{(i)}-u\|_A^2.$$

Proof Denoting by $V_a(b) = \frac{1}{2}\|b-a\|_A^2$ as a function of $b \in \Delta_{\mathsf{box}}$ parameterized at $a \in \Delta_{\mathsf{box}}$, we have $\nabla_i V_a(b) = \|A_{:i}\|_\infty \cdot (a_i - b_i)$. In the optimization language, $V_a(b)$ is the Bregman divergence of the $\|\cdot\|_A^2$ regularizer [12]. We derive the following sequence of inequalities:

$$\begin{aligned}\left\langle n\alpha_k\xi_k^{(i)}, \mathsf{z}_{k-1}-u\right\rangle &= \left\langle n\alpha_k\xi_k^{(i)}, \mathsf{z}_{k-1}-\mathsf{z}_k^{(i)}\right\rangle + \left\langle n\alpha_k\xi_k^{(i)}, \mathsf{z}_k^{(i)}-u\right\rangle \\ &\overset{①}{\le} \left\langle n\alpha_k\xi_k^{(i)}, \mathsf{z}_{k-1}-\mathsf{z}_k^{(i)}\right\rangle + \left\langle -\nabla V_{\mathsf{z}_{k-1}}(\mathsf{z}_k^{(i)}), \mathsf{z}_k^{(i)}-u\right\rangle\end{aligned}$$

$$\overset{②}{=} \langle n\alpha_k \xi_k^{(i)}, \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)} \rangle - \frac{1}{2}\|\mathsf{z}_{k-1} - \mathsf{z}_k^{(i)}\|_A^2 + \frac{1}{2}\|\mathsf{z}_{k-1} - u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k^{(i)} - u\|_A^2$$

$$\overset{③}{=} n^2\alpha_k^2 L\left(\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k \rangle - \frac{L}{2}\|\mathsf{x}_k - \mathsf{y}_k\|_A^2\right) + \frac{1}{2}\|\mathsf{z}_{k-1} - u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k^{(i)} - u\|_A^2$$

$$\leq n^2\alpha_k^2 L \cdot \langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k \rangle + \frac{1}{2}\|\mathsf{z}_{k-1} - u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k^{(i)} - u\|_A^2.$$

Above, ① is due to the minimality of $\mathsf{z}_k^{(i)} = \operatorname{argmin}_{z \in \Delta_{\mathsf{box}}} \{V_{\mathsf{z}_{k-1}}(z) + \langle n\alpha_k \xi_k^{(i)}, z \rangle\}$, which implies that $\langle \nabla V_{\mathsf{z}_{k-1}}(\mathsf{z}_k^{(i)}) + n\alpha \xi_k^{(i)}, u - \mathsf{z}_k^{(i)} \rangle \geq 0$ for all $u \in \Delta_{\mathsf{box}}$. Equality ② can be checked for every coordinate $\ell \in [n]$ as follows:

$$-\nabla_\ell V_{\mathsf{z}_{k-1}}(\mathsf{z}_k^{(i)}) \cdot (\mathsf{z}_{k,\ell}^{(i)} - u_\ell) = \|A_{:i}\|_\infty (\mathsf{z}_{k-1,\ell} - \mathsf{z}_{k,\ell}^{(i)}) \cdot (\mathsf{z}_{k,\ell}^{(i)} - u_\ell)$$
$$= \|A_{:i}\|_\infty \left(-\frac{1}{2}(\mathsf{z}_{k-1,\ell} - \mathsf{z}_{k,\ell}^{(i)})^2 + \frac{1}{2}(u_\ell - \mathsf{z}_{k-1,\ell})^2 - \frac{1}{2}(\mathsf{z}_{k,\ell}^{(i)} - u_\ell)^2\right).$$

③ is by our choice of y_k which satisfies that $\mathsf{z}_{k-1} - \mathsf{z}_k^{(i)} = n\alpha_k L(\mathsf{x}_k - \mathsf{y}_k^{(i)})$. □

In addition, as a simple corollary of Proposition 3.2, we have the following fact

Fact 3.8 $|\mathsf{z}_{k,i}^{(i)} - \mathsf{z}_{k-1,i}| \leq \frac{n\alpha_k |\xi_{k,i}^{(i)}|}{\|A_{:i}\|_\infty}$ *and* $|\mathsf{y}_{k,i}^{(i)} - \mathsf{x}_{k,i}| = \frac{1}{n\alpha_k L}|\mathsf{z}_{k,i}^{(i)} - \mathsf{z}_{k-1,i}| \leq \frac{|\xi_{k,i}^{(k)}|}{L\|A_{:i}\|_\infty} \leq \frac{1}{L\|A_{:i}\|_\infty}$. *If* $\xi_{k,i}^{(i)} \geq 0$, *then* $\mathsf{z}_{k,i}^{(i)} \leq \mathsf{z}_{k-1,i}$ *and* $\mathsf{y}_{k,i}^{(i)} \leq \mathsf{x}_{k,i}$; *if* $\xi_{k,i}^{(i)} \leq 0$, *then* $\mathsf{z}_{k,i}^{(i)} \geq \mathsf{z}_{k-1,i}$ *and* $\mathsf{y}_{k,i}^{(i)} \geq \mathsf{x}_{k,i}$.

3.3 Step 2: gradient descent guarantee

We call our update $\mathsf{y}_k^{(i)} \leftarrow \mathsf{x}_k + \frac{1}{n\alpha_k L}(\mathsf{z}_k^{(i)} - \mathsf{z}_{k-1})$ a gradient descent step, because the following lemma guarantees $f_\mu(\mathsf{y}_k^{(i)}) \leq f_\mu(\mathsf{x}_k)$, that is, the objective only decreases; moreover, the objective decreases at least by $\frac{1}{2}\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \rangle$.

Lemma 3.9 (gradient descent) *We have* $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \geq \frac{1}{2}\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \rangle \geq 0$.

This Lemma 3.9, which is characteristic of the PC-LP setting, is strong in the following sense. Even though the update $y_k^{(i)}$ only depends on the truncated gradient $\xi_k^{(i)}$, the progress we make is a function of the true gradient $\nabla_i f_\mu(\mathsf{x}_k)$, including the large component that was discarded. This is possible because the smoothness guarantee of Lemma 2.7.b allows us to take a long coordinate step even though $f_\mu(x)$ is not Lipschitz-smooth.

Proof of Lemma 3.9 Using Fact 3.8, we write $\mathsf{y}_k^{(i)} = \mathsf{x}_k + s\lambda \mathbf{e}_i$ for some $s \in \{-1, +1\}$ and step length $\lambda \in \left[0, \frac{1}{L\|A_{:i}\|_\infty}\right]$. We first focus on the case $\nabla_i f_\mu(\mathsf{x}_k) \in [-1, 1]$ so $\xi_{k,i}^{(i)} = \nabla_i f_\mu(\mathsf{x}_k)$.

$$\begin{aligned} f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) &= f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{x}_k + s\lambda \mathbf{e}_i) = -s\int_0^\lambda \left(\nabla_i f_\mu(\mathsf{x}_k + s\chi \mathbf{e}_i)\right) d\chi \\ &\overset{①}{\geq} s\int_0^\lambda \left(-\nabla_i f_\mu(\mathsf{x}_k) - L\|A_{:i}\|_\infty \cdot s\chi\right) d\chi = -\nabla_i f_\mu(\mathsf{x}_k) \cdot s\lambda - \frac{L\|A_{:i}\|_\infty}{2} \cdot \lambda^2 \\ &\overset{②}{\geq} -\nabla_i f_\mu(\mathsf{x}_k) \cdot s\lambda - \frac{L\|A_{:i}\|_\infty}{2} \cdot \lambda \cdot \frac{|\xi_{k,i}^{(k)}|}{L\|A_{:i}\|_\infty} \overset{③}{=} -\frac{1}{2}\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{y}_k^{(i)} - \mathsf{x}_k\rangle. \end{aligned}$$

Above, ① uses Lemma 2.7.a, ② uses Fact 3.8, and ③ uses $|\xi_{k,i}^{(k)}| = -s\nabla_i f_\mu(\mathsf{x}_k)$ (see also Fact 3.8). Next, we turn to the case of $\nabla_i f_\mu(\mathsf{x}_k) > 1$. In this case, we have $s = -1$ and

$$\begin{aligned} f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) &= f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{x}_k - \lambda \mathbf{e}_i) = \int_0^\lambda \nabla_i f_\mu(\mathsf{x}_k - \chi \mathbf{e}_i) d\chi \\ &\overset{①}{\geq} \int_0^\lambda \left(1 - \frac{\|A_{:i}\|_\infty L}{2}\chi\right) \nabla_i f_\mu(x) d\chi \\ &\overset{②}{\geq} \int_0^\lambda \frac{1}{2}\nabla_i f_\mu(x) d\chi = \frac{1}{2}\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)}\rangle. \end{aligned}$$

Above, ① uses Lemma 2.7.b and ② uses $\chi \leq \lambda \leq \frac{1}{L\|A_{:i}\|_\infty}$. Finally, we have $\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)}\rangle \geq 0$ because $\nabla_i f_\mu(\mathsf{x}_k)$ and $\mathsf{x}_{k,i} - \mathsf{y}_{k,i}^{(i)}$ have the same sign, and $\mathsf{x}_{k,\ell} = \mathsf{y}_{k,\ell}^{(i)}$ for $\ell \neq i$.

3.4 Step 3: putting all together

We denote by $\eta_k^{(i)} \in \mathbb{R}_{\geq 0}^n$ the vector that is only non-zero at coordinate i, and satisfies $\eta_{k,i}^{(i)} = \nabla_i f_\mu(\mathsf{x}_k) - \xi_{k,i}^{(i)} \in [0, \infty)$. In other words, the full gradient

$$\nabla f_\mu(\mathsf{x}_k) = \mathbf{E}_i[(0, \ldots, n\nabla_i f_\mu(\mathsf{x}_k), \ldots, 0)] = \mathbf{E}_i[n\eta_k^{(i)} + n\xi_k^{(i)}]$$

can be (in expectation) decomposed into the a large non-negative component $\eta_k^{(i)} \in [0, \infty)^n$ and a truncated component $\xi_k^{(i)} \in [-1, 1]^n$. Recall that $\eta_k^{(i)}$ did not contribute to the descent steps (see Line 9 of *PacLPSolver*). Now, for any $u \in \Delta_{\mathsf{box}}$, we can use a basic convexity argument and the mirror descent lemma to compute that

$$\begin{aligned} &\alpha_k(f_\mu(\mathsf{x}_k) - f_\mu(u)) \leq \left\langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - u\right\rangle \\ &= \left\langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{z}_{k-1}\right\rangle + \left\langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{z}_{k-1} - u\right\rangle \end{aligned}$$

$$
\begin{aligned}
&= \left\langle \alpha_k \nabla f_\mu(\mathbf{x}_k), \mathbf{x}_k - \mathbf{z}_{k-1} \right\rangle + \mathbf{E}_i \left[\left\langle n\alpha_k \eta_k^{(i)}, \mathbf{z}_{k-1} - u \right\rangle + \left\langle n\alpha_k \xi_k^{(i)}, \mathbf{z}_{k-1} - u \right\rangle \right] \\
&\overset{①}{=} \frac{(1-\tau)\alpha_k}{\tau} \left\langle \nabla f_\mu(\mathbf{x}_k), \mathbf{y}_{k-1} - \mathbf{x}_k \right\rangle \\
&\quad + \mathbf{E}_i \left[\left\langle n\alpha_k \eta_k^{(i)}, \mathbf{z}_{k-1} - u \right\rangle + \left\langle n\alpha_k \xi_k^{(i)}, \mathbf{z}_{k-1} - u \right\rangle \right] \qquad (3.1) \\
&\overset{②}{\leq} \frac{(1-\tau)\alpha_k}{\tau} \left(f_\mu(\mathbf{y}_{k-1}) - f_\mu(\mathbf{x}_k) \right) \\
&\quad + \mathbf{E}_i \Big[\boxed{\left\langle n\alpha_k \eta_k^{(i)}, \mathbf{z}_{k-1} - u \right\rangle + n^2 \alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathbf{x}_k - \mathbf{y}_k^{(i)} \right\rangle} \\
&\quad + \frac{1}{2} \|\mathbf{z}_{k-1} - u\|_A^2 - \frac{1}{2} \|\mathbf{z}_k^{(i)} - u\|_A^2 \Big] \qquad (3.2)
\end{aligned}
$$

Above, ① is because $\mathbf{x}_k = \tau \mathbf{z}_{k-1} + (1-\tau)\mathbf{y}_{k-1}$, which implies that $\tau(\mathbf{x}_k - \mathbf{z}_{k-1}) = (1-\tau)(\mathbf{y}_{k-1} - \mathbf{x}_k)$. ② uses convexity and Lemma 3.7. This above computation is motivated by [5], and as we shall see below, it allows one to linearly couple gradient and mirror steps.

Intuitively, the first term in the box of (3.2) is the loss introduced by the large gradient $\eta_k^{(i)}$. This part was truncated so did not contribute to the mirror step. The second term in the box is the loss introduced by mirror descent on the small gradient $\xi_k^{(i)}$ in Lemma 3.7.

Now comes an important observation. As shown by Lemma 3.10 below, the performance of the gradient step—that is, the objective decrease of $f_\mu(\mathbf{x}_k) - f_\mu(\mathbf{y}_k^{(i)})$—is at least proportional to the total loss incurred in the box. Intuitively, this means that the progress in the gradient step is so large that it outweighs not only the loss from mirror descent (as is typical in accelerated gradient analyses [5,29]) but also the loss term introduced by $\eta_k^{(i)}$.

Lemma 3.10 (gradient descent total guarantee) *For every $u \geq 0$,*

$$
\left\langle n\alpha_k \eta_k^{(i)}, \mathbf{z}_{k-1} - u \right\rangle + n^2 \alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathbf{x}_k - \mathbf{y}_k^{(i)} \right\rangle \leq 3n\alpha_k L \cdot \left(f_\mu(\mathbf{x}_k) - f_\mu(\mathbf{y}_k^{(i)}) \right).
$$

The proof of Lemma 3.10 is a careful case analysis and several simple applications of Lemma 3.9. We remark that to properly upper bound $\langle n\alpha_k \eta_k^{(i)}, \mathbf{z}_{k-1} - u \rangle$, one needs to have some good upper bound the coordinates of $\mathbf{z}_{k-1}$. This is exactly the place we need our redundant constraint which ensures $\mathbf{z}_{k-1,i} \leq \frac{1}{\|A_{:i}\|_\infty}$ (see Remark 2.5).

Proof of Lemma 3.10 There are three possibilities:

– If $\eta_{k,i}^{(i)} = 0$, then we must have $\xi_{k,i}^{(i)} = \nabla_i f_\mu(\mathbf{x}_k) \in [-1, 1]$. Lemma 3.9 implies

$$
\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathbf{z}_{k-1} - u \right\rangle + n^2 \alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathbf{x}_k - \mathbf{y}_k^{(i)} \right\rangle \\
&\quad = n^2 \alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathbf{x}_k), \mathbf{x}_k - \mathbf{y}_k^{(i)} \right\rangle \leq 2n^2 \alpha_k^2 L \cdot \left(f_\mu(\mathbf{x}_k) - f_\mu(\mathbf{y}_k^{(i)}) \right)
\end{aligned}
$$

– If $\eta_{k,i}^{(i)} > 0$ and $\mathsf{z}_{k,i}^{(i)} > 0$, then we precisely have $\mathsf{z}_{k,i}^{(i)} = \mathsf{z}_{k-1,i} - \frac{n\alpha_k}{\|A_{:i}\|_\infty}$ (see Proposition 3.2), and accordingly $\mathsf{y}_{k,i}^{(i)} = \mathsf{x}_{k,i} - \frac{1}{L\|A_{:i}\|_\infty} < \mathsf{x}_{k,i}$. In this case,

$$\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{①}{\le} n\alpha_k \cdot \nabla_i f_\mu(\mathsf{x}_k) \cdot \frac{1}{\|A_{:i}\|_\infty} + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{②}{<} n\alpha_k \cdot \nabla_i f_\mu(\mathsf{x}_k) \cdot \frac{1}{\|A_{:i}\|_\infty} + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{③}{=} n\alpha_k L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{④}{\le} \left(2n\alpha_k L + 2n^2\alpha_k^2 L\right) \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})).
\end{aligned}$$

Above, ① follows from the fact that $\mathsf{z}_{k-1} \in \Delta_{\mathsf{box}}$ and therefore $\mathsf{z}_{k-1,i} \le \frac{1}{\|A_{:i}\|_\infty}$ by the definition of Δ_{box}, and $u \ge 0$; ② follows from the fact that x_k and $\mathsf{y}_k^{(i)}$ are only different at coordinate i, and $\xi_{k,i}^{(i)} = 1 < \nabla_i f_\mu(\mathsf{x}_k)$ (since $\eta_{k,i}^{(i)} > 0$); ③ follows from the fact that $\mathsf{y}_k^{(i)} = \mathsf{x}_k - \frac{\mathbf{e}_i}{L\|A_{:i}\|_\infty}$; and ④ uses Lemma 3.9.

– If $\eta_{k,i}^{(i)} > 0$ and $\mathsf{z}_{k,i}^{(i)} = 0$, then we have

$$\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{①}{\le} \left(n\alpha_k \nabla_i f_\mu(\mathsf{x}_k) \cdot \mathsf{z}_{k-1,i}\right) + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{②}{=} \left\langle n\alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)} \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{③}{=} n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{④}{\le} 4n^2\alpha_k^2 L \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})).
\end{aligned}$$

Above, ① is because $u \ge 0$, $\nabla_i f_\mu(\mathsf{x}_k) = \eta_{k,i}^{(i)} + 1 > \eta_{k,i}^{(i)}$ and $\nabla_i f_\mu(\mathsf{x}_k) > \xi_{k,i}^{(i)}$; ② uses the assumption that $\mathsf{z}_{k,i}^{(i)} = 0$ and the fact that $\mathsf{z}_{k-1,\ell} = \mathsf{z}_{k,\ell}^{(i)}$ for every $\ell \ne i$; ③ is from our choice of y_k which satisfies that $\mathsf{z}_{k-1} - \mathsf{z}_k^{(i)} = n\alpha_k L(\mathsf{x}_k - \mathsf{y}_k^{(i)})$; and ④ uses Lemma 3.9.

Combining the three cases, and using the fact that $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge 0$, we conclude that

$$\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\quad \le (2n\alpha_k L + 4n^2\alpha_k^2 L) \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})) \\
&\quad \le 3n\alpha_k L \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})).
\end{aligned}$$

Above, the last inequality uses our choice of α_k, which implies $n\alpha_k \le n\alpha_T = \frac{1}{\varepsilon L} \le \frac{1}{4}$.

Plugging Lemma 3.10 back to (3.2), we have

$$\begin{aligned}\alpha_k(f_\mu(\mathsf{x}_k)-f_\mu(u)) &\le \big\langle \alpha_k\nabla f_\mu(\mathsf{x}_k),\mathsf{x}_k-u\big\rangle\\ &\overset{①}{\le} \frac{(1-\tau)\alpha_k}{\tau}(f_\mu(\mathsf{y}_{k-1})-f_\mu(\mathsf{x}_k))\\ &\quad+\mathbf{E}_i\Big[3n\alpha_k L\cdot(f_\mu(\mathsf{x}_k)-f_\mu(\mathsf{y}_k^{(i)}))+\frac{1}{2}\|\mathsf{z}_{k-1}-u\|_A^2-\frac{1}{2}\|\mathsf{z}_k-u\|_A^2\Big]\\ &\overset{②}{\le} \alpha_k f_\mu(\mathsf{x}_k)+\big(3n\alpha_k L-\alpha_k\big)f_\mu(\mathsf{y}_{k-1})\\ &\quad+\mathbf{E}_i\Big[-3n\alpha_k L\cdot f_\mu(\mathsf{y}_k^{(i)})+\frac{1}{2}\|\mathsf{z}_{k-1}-u\|_A^2-\frac{1}{2}\|\mathsf{z}_k-u\|_A^2\Big].\end{aligned}\tag{3.3}$$

Above, ① uses Lemma 3.10; and ② is because we have chosen τ to satisfy $\frac{1}{\tau}=3nL$.

Next, recall that we have picked α_k so that $(3nL-1)\alpha_k=3nL\cdot\alpha_{k-1}$ in Algorithm 1. Telescoping (3.3) for $k=1,\dots,T$ and choosing $u^*=(1-\varepsilon/2)x^*$, we have

$$\begin{aligned}-\sum_{k=1}^{T}\alpha_k f_\mu(u^*) &\le 3f_\mu(\mathsf{y}_0)-3n\alpha_T L\cdot\mathbf{E}[f_\mu(\mathsf{y}_T)]+\|\mathsf{z}_0-u^*\|_A^2\\ &\le -3n\alpha_T L\cdot\mathbf{E}[f_\mu(\mathsf{y}_T)]+\mathsf{OPT}.\end{aligned}$$

Here, the second inequality is due to $f_\mu(\mathsf{y}_0)=f_\mu(x^{\mathsf{start}})\le 0$ from Fact 2.8, and the fact that

$$\|\mathsf{z}_0-u^*\|_A^2=\|u^*\|_A^2=\sum_{i=1}^{n}(u_i^*)^2\cdot\|A_{:i}\|_\infty\le\sum_{i=1}^{n}(x_i^*)^2\cdot\|A_{:i}\|_\infty\le\sum_{i=1}^{n}x_i^*=\mathsf{OPT}.$$

Finally, using the fact that $\sum_{k=1}^{T}\alpha_k=\alpha_T\cdot\sum_{k=0}^{T-1}\big(1-\frac{1}{3nL}\big)^k=3n\alpha_T L\big(1-(1-\frac{1}{3nL})^T\big)$, we rearrange and obtain that

$$\begin{aligned}\mathbf{E}[f_\mu(\mathsf{y}_T)] &\le \frac{\sum_k\alpha_k}{3n\alpha_T L}f_\mu(u^*)+\frac{1}{3n\alpha_T L}\mathsf{OPT}=\Big(1-\Big(1-\frac{1}{3nL}\Big)^T\Big)f_\mu(u^*)\\ &\quad+\frac{1}{3n\alpha_T L}\mathsf{OPT}.\end{aligned}$$

We choose $T=\lceil 3nL\log(1/\varepsilon)\rceil$ so that $\frac{1}{n\alpha_T L}=(1-\frac{1}{3nL})^T\le\varepsilon$. Combining this with the fact that $f_\mu(u^*)\le-(1-\varepsilon)\mathsf{OPT}<0$ (see Proposition 2.4.a), we obtain

$$\mathbf{E}[f_\mu(\mathsf{y}_T)]\le(1-\varepsilon)f_\mu(u^*)+\varepsilon/3\cdot\mathsf{OPT}<-(1-3\varepsilon)\mathsf{OPT}.$$

Therefore, we have finished proving Theorem 3.4. □

4 Relaxation of the covering linear program

Since *PacLPSolver* gives only an approximate packing LP solution, we cannot infer from it a dual covering LP solution. Therefore, we have to work on a relaxed version of covering LP directly. For input matrix $A \in \mathbb{R}^{m\times n}_{\geq 0}$, we rewrite the covering LP problem (1.2) as follows in order to be notationally close to packing LP:

$$\min_{x\geq 0}\{\mathbb{1}^T x \ : \ Ax \geq \mathbb{1}\}. \tag{4.1}$$

We denote by OPT the optimal value to this LP, and by x^* any of its optimal solutions. We say that x is a $(1+\varepsilon)$-approximation for the covering LP if $Ax \geq \mathbb{1}$ and $\mathbb{1}^T x \leq (1+\varepsilon)\mathsf{OPT}$.

Again, we use indices $i \in [n]$ for the columns of A, and indices $j \in [m]$ for the rows of A. We denote by $A_{:i}$ the i-th column vector of A, and $A_{j:}$ the j-th row vector of A. We assume without loss of generality by simple scaling that [10]

$$\min_{j\in[m]}\{\|A_{j:}\|_\infty\} = 1. \tag{4.2}$$

Proposition 4.1 *The normalization* (4.2) *ensures* $\mathsf{OPT} \in [1, m]$.

Proof Suppose that j^* is the row that achieves the smallest infinite norm $\|A_{j:}\|_\infty$ over all rows $j \in [m]$. Then, for any solution $x \in \mathbb{R}^n_{\geq 0}$ satisfying $\langle A_{j^*:}, x\rangle \geq 1$, we must have $\mathbb{1}^T x \geq 1/\|A_{j^*:}\|_\infty = 1$ using (4.2). On the other hand, we can construct a feasible solution x as follows. Initialize $x = 0$, and then for each row j, let us find the coordinate i that maximizes the value of A_{ij} among all columns i. Then, we increase x_i by $1/A_{ij} = 1/\|A_{j:}\|_\infty$. After we have exhausted all the m rows, we arrive at some $x \geq 0$ satisfying $Ax \geq \mathbb{1}$ as well as $\mathbb{1}^T x = \sum_j 1/\|A_{j:}\|_\infty \leq m$. □

In our covering LP solvers, we assume that an initial solution, achieving a constant approximation, is available to the algorithm. Such a solution can be obtained for instance by the covering LP solver from Young [36] with constant ϵ in time $O(N \log N)$.

Definition 4.2 Let $x^\sharp$ be a given 2-approximate solution to the covering problem given and let $\mathsf{OPT}' \stackrel{\text{def}}{=} \mathbb{1}^T x^\sharp \in [\mathsf{OPT}, 2\mathsf{OPT}]$. Without loss of generality, assume $\mathsf{OPT}' \geq 2$.

We now introduce the smoothed objective $f_\mu(x)$ we are going to minimize in order to solve covering LP. Symmetric to the case in the packing solver, this smoothed objective $f_\mu(x)$ turns each row of the LP constraint $Ax \geq \mathbb{1}$ into an exponential penalty function.

[10] If $\min_{j\in[m]}\{\|A_{j:}\|_\infty\} = 0$ then the covering LP is infeasible so we are done. Otherwise, if $\min_{j\in[m]}\{\|A_{j:}\|_\infty\} = v > 0$ we scale all entries of A by $1/v$, and scale OPT by v.

Definition 4.3 Letting $\mu \stackrel{\text{def}}{=} \frac{\varepsilon}{4\log(nm/\varepsilon)}$, we define the smoothed objective $f_\mu(x)$ as

$$f_\mu(x) \stackrel{\text{def}}{=} \mu \sum_{j=1}^m e^{\frac{1}{\mu}(1-(Ax)_j)} + \mathbb{1}^T x.$$

Fact 4.4 *$\nabla f_\mu(x) = \mathbb{1} - A^T p(x)$ and $\nabla^2 f_\mu(x) = \frac{1}{\mu} A^T \mathrm{diag}\{p(x)\} A$, where $p_j(x) \stackrel{\text{def}}{=} e^{\frac{1}{\mu}(1-(Ax)_j)}$.*

We present some properties about $f_\mu(x)$. They together imply that the minimum of $f_\mu(x)$ is around OPT, and if one approximately finds the minimum of $f_\mu(x)$ up to an additive error $O(\varepsilon\mathsf{OPT})$, this corresponds to a $(1+O(\varepsilon))$-approximate solution to the covering LP (4.1). The proofs are analogous to Sect. 2, and included in "Appendix B.2" for completeness' sake.

Proposition 4.5 (a) *$f_\mu(u^*) \le (1+\varepsilon)\mathsf{OPT}$ for $u^* \stackrel{\text{def}}{=} (1+\varepsilon/2)x^*$.*
(b) *$f_\mu(x) \ge (1-\varepsilon)\mathsf{OPT}$ for every $x \ge 0$.*
(c) *For any $x \ge 0$ satisfying $f_\mu(x) \le 2\mathsf{OPT}$, we must have $Ax \ge (1-\varepsilon)\mathbb{1}$.*
(d) *If $x \ge 0$ satisfies $f_\mu(x) \le (1+\delta)\mathsf{OPT}$ for some $\delta \in [0,1]$, then $\frac{1}{1-\varepsilon}x$ is a $\frac{1+\delta}{1-\varepsilon}$-approximate solution to the covering LP.*

5 Our covering LP solver in the well-conditioned case

Recall in packing LPs, since it satisfies $0 \le x_i^* \le \frac{1}{\|A_{:i}\|_\infty}$ (see Fact 2.1), we can minimize f_μ over a bounding box Δ_{box}. Unfortunately, it no longer satisfies $x_i^* \le \frac{1}{\|A_{:i}\|_\infty}$ in covering LPs, so one cannot directly turn *PacLPSolver* into its symmetric version to solve covering LP.

In this section, we show that this symmetric covering LP solver still solves all *well-behaved* covering LP instances. Specifically, we say the covering LP is well-behaved if:[11]

Assumption 5.1 There exists some optimal covering LP solution x^* satisfying $x_i^* \le \frac{9}{\|A_{:i}\|_\infty}$; and the initial point $x^\sharp$ satisfies $x_i^\sharp \le \frac{9}{\|A_{:i}\|_\infty}$.

For instance, well-behaved instances naturally arise from those where the constraints $Ax \ge \mathbb{1}$ are non-redundant. If the optimal covering LP solution x^* and the initial point $x^\sharp$ satisfy $\mathbb{1} \le Ax^* \le 9 \cdot \mathbb{1}$ and $\mathbb{1} \le Ax^\sharp \le 9 \cdot \mathbb{1}$, then Assumption 5.1 is satisfied.

Well-behaved covering LP problems immediately satisfy the following:

Fact 5.2 *Define $\Delta_{\mathsf{box}} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : x_i \in \left[0, \frac{10}{\|A_{:i}\|_\infty}\right]\}$. Under Assumption 5.1, we have $u^* \stackrel{\text{def}}{=} (1+\varepsilon/2)x^* \in \Delta_{\mathsf{box}}$ and $x^{\mathsf{start}} \stackrel{\text{def}}{=} (1+\varepsilon/2)\cdot x^\sharp \in \Delta_{\mathsf{box}}$. Also, it satisfies $f_\mu(x^{\mathsf{start}}) \le 3\mathsf{OPT}$.*

[11] The constant 9 in this section can be replaced with any other constant greater than 1.

Algorithm 2 $CovLPSolver^{\mathsf{wb}}(A, x^{\mathsf{start}}, \varepsilon)$

Input: $A \in \mathbb{R}^{m\times n}_{\geq 0}$, $x^{\mathsf{start}} \in \Delta_{\mathsf{box}}$, $\varepsilon \in (0, 1/30]$.

Output: $x \in \Delta_{\mathsf{box}}$. ▷ recall $\Delta_{\mathsf{box}} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : x_i \in [0, \frac{10}{\|A_{:i}\|_\infty}]\}$

1: $\mu \leftarrow \frac{\varepsilon}{4\log(nm/\varepsilon)}$, $L \leftarrow \frac{4}{\mu}$, $\tau \leftarrow \frac{1}{21\cdot nL}$ and $\alpha_0 \leftarrow \frac{1}{nL}$. ▷ parameters

2: $T \leftarrow \lceil 21nL\log(1/\varepsilon)\rceil = O(n\cdot\frac{\log(nm/\varepsilon)\cdot\log(1/\varepsilon)}{\varepsilon})$. ▷ number of iterations

3: $\mathsf{x}_0 = \mathsf{y}_0 \leftarrow x^{\mathsf{start}}$, $\mathsf{z}_0 \leftarrow 0$.

4: **for** $k \leftarrow 1$ **to** T **do**

5: $\quad\alpha_k \leftarrow \frac{1}{1-\tau}\alpha_{k-1}$

6: $\quad\mathsf{x}_k \leftarrow \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$.

7: $\quad$Randomly select $i \in [n]$ uniformly at random.

8: $\quad$Define vector $\xi_k^{(i)}$ be all-zero except at coordinate i, where $\xi_{k,i}^{(i)} = \max\{-1, \nabla_i f_\mu(\mathsf{x}_k))\}$.

9: $\quad\mathsf{z}_k \leftarrow \mathsf{z}_k^{(i)} \stackrel{\text{def}}{=} \text{argmin}_{z\in\Delta_{\mathsf{box}}}\left\{\frac{1}{2}\|z - \mathsf{z}_{k-1}\|_A^2 + \langle n\alpha_k\xi_k^{(i)}, z\rangle\right\}$. ▷ See Proposition 3.2

10: $\quad\mathsf{y}_k \leftarrow \mathsf{y}_k^{(i)} \stackrel{\text{def}}{=} \mathsf{x}_k + \frac{1}{n\alpha_k L}(\mathsf{z}_k^{(i)} - \mathsf{z}_{k-1})$.

11: **end for**

12: **return** y_T.

Proof The claims $u^*, x^{\mathsf{start}} \in \Delta_{\mathsf{box}}$ are trivial after noticing $\varepsilon \leq 1/30$. Using the fact that $Ax^{\mathsf{start}} - \mathbb{1} \geq (1+\varepsilon/2)Ax^\sharp - \mathbb{1} \geq \varepsilon/2\cdot\mathbb{1}$, we compute $f_\mu(x^{\mathsf{start}})$ as follows:

$$f_\mu(x^{\mathsf{start}}) = \mu\sum_j e^{\frac{1}{\mu}(1-(Ax^{\mathsf{start}})_j)} + \mathbb{1}^T x^{\mathsf{start}} \leq \mu\sum_j e^{\frac{-\varepsilon/2}{\mu}} + 2\mathsf{OPT}$$
$$\leq \frac{\mu m}{(nm)^2} + 2\mathsf{OPT} < 3\mathsf{OPT}.$$

□

We now describe $CovLPSolver^{\mathsf{wb}}$ (which is a symmetric variant of $PacLPSolver$) that solves well-behaved covering LP problems, see Algorithm 2. It starts with the initial vector $\mathsf{x}_0 = \mathsf{y}_0 = x^{\mathsf{start}}$ and $\mathsf{z}_0 = 0$. Then, $CovLPSolver^{\mathsf{wb}}$ is divided into T iterations. In each iteration k, it computes a weighted midpoint $\mathsf{x}_k \leftarrow \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$ for some parameter $\tau \in (0,1)$, and then proceeds to compute y_k and z_k as follows.

We select $i \in [n]$ uniformly at random. Let $\xi_k^{(i)} = (0, \ldots, 0, -\mathbb{T}^{\mathsf{p}}(v), 0, \ldots, 0)$ be the vector that is only non-zero at coordinate i, where $v = -\nabla_i f_\mu(\mathsf{x}_k) \in [-1, \infty)$, and recall $\mathbb{T}^{\mathsf{p}}(v) \stackrel{\text{def}}{=} \min\{v, 1\}$. We refer to $\xi_k^{(i)}$ as the *truncated gradient*. Next,

- Perform a *mirror (descent) step* $\mathsf{z}_k \leftarrow \mathsf{z}_k^{(i)} \stackrel{\text{def}}{=} \text{argmin}_{z\in\Delta_{\mathsf{box}}}\left\{\frac{1}{2}\|z - \mathsf{z}_{k-1}\|_A^2 + \langle n\alpha_k\xi_k^{(i)}, z\rangle\right\}$ for some parameter $\alpha_k \ll 1/n$ to be chosen later.
- Perform a *gradient (descent) step* $\mathsf{y}_k \leftarrow \mathsf{y}_k^{(i)} \stackrel{\text{def}}{=} \mathsf{x}_k + \frac{1}{n\alpha_k L}(\mathsf{z}_k^{(i)} - \mathsf{z}_{k-1})$.

This finishes the description of $CovLPSolver^{\mathsf{wb}}$. It is not surprising to deduce the following theorem similar to Theorem 3.4. We include its proof in "Appendix C.3" for completeness.

Theorem 5.3 *Under the well-behavior assumption 5.1 in the covering LP problem,* $CovLPSolver^{\mathsf{wb}}(A, x^{\mathsf{start}}, \varepsilon)$ *outputs some* y_T *satisfying*

$$\mathbf{E}[f_\mu(\mathsf{y}_T)] \le (1 + 4.6\varepsilon)\mathsf{OPT}.$$

Again, using the same proof as Corollary 3.5, one can apply Markov's bound to turn Theorem 5.3 into a probabilistic statement:

Corollary 5.4 *Under the well-behavior assumption 5.1 in the covering LP problem, with probability at least* $2/3$, $\mathsf{y}_T = CovLPSolver^{\mathsf{wb}}(A, x^{\mathsf{start}}, \varepsilon)$ *satisfies that* $\frac{\mathsf{y}_T}{1-\varepsilon}$ *is a* $(1+O(\varepsilon))$ *approximate solution to covering LP. The expected running time is*

$$O\left(\frac{\log(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon}N\right).$$

Removing the well-behavior assumption In subsequent work, after the conference presentation of this paper, Wang, Mahoney and Rao [34] showed the following theorem.

Theorem 5.5 *([34]) Any covering LP with constraint matrix* $A \in \mathbb{R}^{m\times n}_{\ge 0}$ *of sparsity* N *can be converted into an equivalent but* well-behaved *covering LP with matrix* $\widetilde{A} \in \mathbb{R}^{m\times n\cdot O(\log(mn/\varepsilon))}$ *and sparsity* $N \cdot O(\log(mn/\varepsilon))$. *The conversion takes time* $N \cdot O(\log(mn/\varepsilon))$.

As a result, we can apply our covering solver $CovLPSolver^{\mathsf{wb}}$ to this modified LP and apply our Theorem 5.3 to solve any covering LP in expected time $O(\frac{\log^2(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon}N)$.

6 Our covering LP solver in the general case

In this section, we remove the well-behavior assumption and propose a different algorithm $CovLPSolver$ to solve all covering LP instances. This algorithm introduces a factor $1/\sqrt{\varepsilon}$ loss in the running time, but is a *direct* covering LP solver without using any reduction.

The main difference to $PacLPSolver$ and $CovLPSolver^{\mathsf{wb}}$ is that, this time we abandon the box constraint and study the minimization of $f_\mu(x)$ over a simplex

$$x \in \Delta_{\mathsf{simplex}} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \,:\, x_i \ge 0 \ \wedge\ \mathbb{1}^T x \le 2\mathsf{OPT}'\}.$$

Again, this constraint $\mathbb{1}^T x \le 2\mathsf{OPT}'$ is redundant just like the old Δ_{box} constraint for packing LP (recall Remark 2.5); however, it shall be used to make sure that our updates are always inside $\Delta_{\mathsf{simplex}}$. It is a simple fact that

Fact 6.1 $u^* \stackrel{\text{def}}{=} (1+\varepsilon/2)x^* \in \Delta_{\mathsf{simplex}}$.

Recall that the initial vector $x^\sharp$ is defined in Definition 4.2, and OPT' is a crude approximation to OPT, satisfying $\mathsf{OPT}' \stackrel{\text{def}}{=} \mathbb{1}^T x^\sharp \in [\mathsf{OPT}, 2\mathsf{OPT}]$. We choose different starting vector x^{start} from Sect. 5:

Algorithm 3 *CovLPSolver*$(A, x^{\mathsf{start}}, \varepsilon)$

Input: $A \in \mathbb{R}^{m\times n}_{\geq 0}$, $x^{\mathsf{start}} \in \Delta_{\mathsf{simplex}}$, $\varepsilon \in (0, 1/30]$.
Output: $x \in \Delta_{\mathsf{simplex}}$.

$\mu \leftarrow \frac{\varepsilon}{4\log(nm/\varepsilon)}$, $\beta \leftarrow \sqrt{\varepsilon}$, $\tau \leftarrow \frac{\mu\beta}{12n}$. ▷ parameters
$T \leftarrow \lceil \frac{1}{\tau}\log(1/\varepsilon)\rceil = O(\frac{\log(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon^{1.5}}n)$. ▷ number of iterations
$\alpha_0 \leftarrow (1-\tau)^T \frac{\varepsilon}{12n\beta}$ and $\gamma \leftarrow \frac{\varepsilon}{6\beta}$. ▷ so that $\alpha_T = \frac{\varepsilon}{12n\beta}$ and $\gamma = 2\alpha_T n$
$\mathsf{x}_0 = \mathsf{y}_0 = \mathsf{z}_0 \leftarrow x^{\mathsf{start}}$.
for $k \leftarrow 1$ **to** T **do**
  $\alpha_k \leftarrow \frac{1}{1-\tau}\alpha_{k-1}$.
  $\mathsf{x}_k \leftarrow \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$.
  Randomly select i uniformly at random from $[n]$.
  Define vector $\xi_k^{(i)}$ to be all-zero except at coordinate i, where $\xi_{k,i}^{(i)} = \max\{-\beta, \nabla_i f_\mu(\mathsf{x}_k)\}$.
  $\mathsf{z}_k \leftarrow \mathsf{z}_k^{(i)} \stackrel{\text{def}}{=} \operatorname{argmin}_{z\in\Delta_{\mathsf{simplex}}}\left\{V_{\mathsf{z}_{k-1}}(z) + \langle(1+\gamma)n\alpha_k\xi_k^{(i)}, z\rangle\right\}$. ▷ See Proposition 6.4
  if $\nabla_i f_\mu(\mathsf{x}_k) < -\beta$ **then**
    Denote by π the permutation that satisfies $A_{\pi(1),i} \leq \cdots \leq A_{\pi(m),i}$.
    Pick $j^* \in [m]$ such that $\begin{cases}\sum_{j<j^*} A_{\pi(j),i}\cdot p_{\pi(j)}(\mathsf{x}_k) < 1+\beta \\ \sum_{j\leq j^*} A_{\pi(j),i}\cdot p_{\pi(j)}(\mathsf{x}_k) \geq 1+\beta\end{cases}$ ▷ $j^* \in [m]$ always exists, see (6.1)
    $\mathsf{y}_k \leftarrow \mathsf{y}_k^{(i)} \stackrel{\text{def}}{=} \mathsf{x}_k + \delta\cdot\mathbf{e}_i$ where $\delta = \frac{\mu\beta}{2A_{\pi(j^*),i}}$.
  else
    $\mathsf{y}_k \leftarrow \mathsf{y}_k^{(i)} \stackrel{\text{def}}{=} \mathsf{x}_k$.
  end if
end for
return y_T.

Proposition 6.2 *Letting* $x^{\mathsf{start}} \stackrel{\text{def}}{=} (1+\varepsilon/2)\cdot x^\sharp + (\frac{1}{n}, \ldots, \frac{1}{n})$, *we have* $x^{\mathsf{start}} \in \Delta_{\mathsf{simplex}}$ *and* $f_\mu(x^{\mathsf{start}}) \leq 4\mathsf{OPT}$.

Proof Using $Ax^{\mathsf{start}} - \mathbb{1} \geq (1+\varepsilon/2)Ax^\sharp - \mathbb{1} \geq \varepsilon/2\cdot\mathbb{1}$, we compute $f_\mu(x^{\mathsf{start}})$ as follows:

$$f_\mu(x^{\mathsf{start}}) = \mu\sum_j e^{\frac{1}{\mu}(1-(Ax^{\mathsf{start}})_j)} + \mathbb{1}^T x^{\mathsf{start}} \leq \mu\sum_j e^{\frac{-\varepsilon/2}{\mu}} + 2\mathsf{OPT} + 1$$
$$\leq \frac{\mu m}{(nm)^2} + 3\mathsf{OPT} < 4\mathsf{OPT}.$$

Also, we have $\mathbb{1}^T x^{\mathsf{start}} \leq (1+\varepsilon/2)\mathsf{OPT}' + 1 \leq 2\mathsf{OPT}'$. (Recall $\mathsf{OPT}' \geq 2$ in Definition 4.2.) □

Our proposed algorithm *CovLPSolver* starts with the initial vector $\mathsf{x}_0 = \mathsf{y}_0 = \mathsf{z}_0 = x^{\mathsf{start}}$ and is divided into T iterations. In each iteration k, as usual, it computes a weighted midpoint $\mathsf{x}_k \leftarrow \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$ for some parameter $\tau \in (0,1)$, and then computes y_k and z_k as follows.

We select $i \in [n]$ uniformly at random, and let $\xi_k^{(i)} = (0, \ldots, 0, \mathbb{T}^{\mathsf{c}}(v), 0, \ldots, 0)$ be the vector that is only non-zero at coordinate i, where $v = \nabla_i f_\mu(\mathsf{x}_k) \in (-\infty, 1]$ and $\mathbb{T}^{\mathsf{c}}(v) \stackrel{\text{def}}{=} \max\{-\beta, v\}$ is the new thresholding function for some parameter $\beta \stackrel{\text{def}}{=} \sqrt{\varepsilon}$. Then,

– Perform a *mirror (descent) step* $\mathsf{z}_k \leftarrow \mathsf{z}_k^{(i)} \stackrel{\text{def}}{=} \text{argmin}_{z \in \Delta_{\text{simplex}}} \left\{ V_{\mathsf{z}_{k-1}}(z) + \langle (1+\gamma) n \alpha_k \xi_k^{(i)}, z \rangle \right\}$ for some positive parameters $\gamma \ll 1$ and $\alpha_k \ll 1/n$, where

$$V_x(y) \stackrel{\text{def}}{=} \sum_{i=1}^n y_i \log \frac{y_i}{x_i} + x_i - y_i$$

is the so-called Bregman divergence of the generalized entropy function (c.f. [12]).

– If $\nabla_i f_\mu(x_k) < -\beta$, perform a *gradient (descent) step* $\mathsf{y}_k \leftarrow \mathsf{y}_k^{(i)} \stackrel{\text{def}}{=} \mathsf{x}_k + \delta \mathbf{e}_i$ for some $\delta > 0$. In practice, one can line-search over δ, but we choose an explicit δ as follows.
  – Denote by π the permutation that satisfies $A_{\pi(1),i} \le \cdots \le A_{\pi(m),i}$
  – Pick $j^* \in [m]$ s.t. $\begin{cases} \sum_{j<j^*} A_{\pi(j),i} \cdot p_{\pi(j)}(\mathsf{x}_k) < 1+\beta \\ \sum_{j \le j^*} A_{\pi(j),i} \cdot p_{\pi(j)}(\mathsf{x}_k) \ge 1+\beta \end{cases}$. Such j^* exists because

$$\sum_{j=1}^m A_{\pi(j),i} \cdot p_{\pi(j)}(\mathsf{x}_k) = \sum_{j=1}^m A_{ji} \cdot p_j(\mathsf{x}_k) = 1 - \nabla_i f_\mu(\mathsf{x}_k) \ge 1+\beta. \tag{6.1}$$

  – Set $\delta = \frac{\mu\beta}{2A_{\pi(j^*),i}}$.

This finishes the description of our *CovLPSolver*.

Remark 6.3 We use the superscript $^{(i)}$ on $\xi_k^{(i)}, \mathsf{y}_k^{(i)}$ and $\mathsf{z}_k^{(i)}$ to emphasize that the value depends on the choice of i. We have used generic parameters τ, α_k, T in the above description and their precise values are presented in Algorithm 3.

Our update on y_k is a "gradient descent step" because we shall prove that it strictly decreases the objective (i.e., $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge 0$). Our update on z_k is a "mirror descent step" because we shall apply standard mirror descent analysis [12] to it. We explicitly describe how to implement this mirror step: (proved in "Appendix D.4")

Proposition 6.4 *If $\mathsf{z}_{k-1} \in \Delta_{\text{simplex}}$ and $\mathsf{z}_{k-1} > 0$, the minimizer $z = \text{argmin}_{z \in \Delta_{\text{simplex}}} \left\{ V_{\mathsf{z}_{k-1}}(z) + \langle \delta \mathbf{e}_i, z \rangle \right\}$ for any scalar $\delta \in \mathbb{R}$ and basis vector $\mathbf{e}_i$ can be computed as follows:*

1. $z \leftarrow \mathsf{z}_{k-1}$.
2. $z_i \leftarrow z_i \cdot e^{-\delta}$.
3. *If* $\mathbb{1}^T z > 2\mathsf{OPT}'$, $z \leftarrow \frac{2\mathsf{OPT}'}{\mathbb{1}^T z} z$.
4. *Return* z.

We also point out that

Lemma 6.5 *Each iteration of CovLPSolver can be implemented to run in expected $O(N/n)$ time. The total expected running time is $O(TN/n)$.*

The proof of Lemma 6.5 is analogous to its packing counterpart, and included in Sect. F.6.

6.1 Main proof ideas and ingredients

In *CovLPSolver*, we pick a random coordinate $i \in [n]$ at each iteration, and decompose $\nabla_i f(\mathsf{x}_k) = \xi + \eta$, where $\eta \in (-\infty, 0]$ is the (negative) large gradient component and $\xi \in [-\sqrt{\varepsilon}, 1]$ is the truncated gradient component. In other words, we truncate the gradient $\nabla_i f(\mathsf{x}_k)$ at a negative threshold $-\beta = -\sqrt{\varepsilon}$, rather than at -1 as in *CovLPSolver*$^{\mathsf{wb}}$.

The reason for this new threshold $-\sqrt{\varepsilon}$ can be understood as follows. In *PacLPSolver* (and symmetrically in *CovLPSolver*$^{\mathsf{wb}}$), we used Lemma 3.10 to show that our gradient descent step y_k decreases the objective by an amount that both includes the ξ and η components. Unfortunately, for covering LP, this decrease amount is only proportional to η but not to ξ (compare Lemma 3.10 with Lemma 6.14 later). This forces us to treat the small gradient ξ separately using mirror descent, but not gradient descent.

If ξ were in $[-1, 1]$, classical theory of mirror descent [12] (or multiplicative weight update [7]) would imply that the mirror step z_k converges at a rate $\propto \varepsilon^{-2}$. This is too slow. Instead, since truncated ξ into a smaller interval $[-\sqrt{\varepsilon}, 1]$, using a *negative-width technique* (see Sect. 6.5), we improve this mirror-descent convergence rate from ε^{-2} to $\varepsilon^{-1.5}$.

On the other hand, due to this truncation at $-\sqrt{\varepsilon}$ instead of -1, our gradient step on y_k also converges slower, at a rate $1/\varepsilon^{1.5}$ instead of $1/\varepsilon$. This is why $\beta = \sqrt{\varepsilon}$ is the best truncation threshold, as it balances gradient and mirror descent.

Another ingredient behind our proof is a new distance bound that is uncommon in first-order analysis. Recall that, given convex function $g(x)$, traditional analysis applies convexity argument $g(x) - g(x^*) \le \langle \nabla g(x), x - x^* \rangle$ to bound the objective distance to optimum. If $g(x) = e^{-x}$ is univariate, $x = -1$, and $x^* = -100$, this bound becomes $e^{-1} \approx e^{-1} - e^{-100} \le e^{-1} \cdot 99$, which is very loose. In our analysis, we replace this convexity argument with a more benign bound, specifically designed for covering LP (see Lemma 6.10).

6.2 Convergence statement

The main convergence theorem of this section is as follows:

Theorem 6.6 *CovLPSolver*$(A, x^{\mathsf{start}}, \varepsilon)$ *outputs some* y_T *satisfying*

$$\mathbf{E}[f_\mu(\mathsf{y}_T)] \le (1 + 9\varepsilon)\mathsf{OPT}.$$

Again, using the same proof as Corollary 3.5, one can apply Markov's bound to turn Theorem 5.3 into a probabilistic statement:

Corollary 6.7 *With probability at least* $2/3$, $\mathsf{y}_T = $ *CovLPSolver*$(A, x^{\mathsf{start}}, \varepsilon)$ *satisfies that* $\frac{\mathsf{y}_T}{1-\varepsilon}$ *is a* $(1 + O(\varepsilon))$ *approximate solution to the covering LP program. The expected running time is* $O(\frac{\log(nm/\varepsilon)\log(1/\varepsilon)}{\varepsilon^{1.5}} N)$.

Before delving into the proof of Theorem 6.6, we make the following observations:

Fact 6.8 *For every* $k \in \{0, 1, \ldots, T\}$, *it satisfies* $\mathsf{x}_k, \mathsf{y}_k \geq 0$, $\mathsf{z}_k > 0$, *and* $\mathsf{z}_k \in \Delta_{\mathsf{simplex}}$.

Proof Since the x^{start} satisfies $\mathbb{1}^T x^{\mathsf{start}} \leq 2\mathsf{OPT}'$ by Proposition 6.2, we have $\mathsf{z}_0 = x^{\mathsf{start}} \in \Delta_{\mathsf{simplex}}$. Also, the mirror descent step (see Proposition 6.4) ensures $\mathsf{z}_{k,i} > 0$ for all rounds k and coordinates i, as well as $\mathsf{z}_k \in \Delta_{\mathsf{simplex}}$ for all rounds k. However, we x_k and y_k may not necessarily lie inside $\Delta_{\mathsf{simplex}}$, but will always stay non-negative. □

We prove Theorem 6.6 in the subsequent subsections.

6.3 Step 1: distance adjustment

Using convexity, one can obtain

$$f_\mu(\mathsf{x}_k) - f_\mu(u) \leq \langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - u \rangle \quad \text{for every } u \in \Delta_{\mathsf{simplex}}. \tag{6.2}$$

Note that inequality (6.2) can be very loose for exponential functions. For instance, if $f_\mu(x)$ were as simple as e^x, then the convexity inequality $e^b - e^a \leq e^b \cdot (b - a)$ says

- when $b = 2$ and $a = -10$, we have $e^2 - e^{-10} \leq 12e^2$;
- when $b = 2$ and $a = -100$, we have $e^2 - e^{-100} \leq 102e^2$.

Although $e^{-100} \approx e^{-10}$, the two upper bounds are off from each other by a factor of 10.

In this section, we strengthen (6.2) in the special case of $u = u^* \stackrel{\text{def}}{=} (1 + \varepsilon/2)x^*$. For analysis purpose, let $\widetilde{A}$ be the *adjusted matrix* of A described as follows.

Definition 6.9 (adjusted matrix $\widetilde{A}$) For each row $j \in [m]$, if $(Au^*)_j \leq 2$ then we keep it and let $\widetilde{A}_{j:} \stackrel{\text{def}}{=} A_{j:}$. Otherwise,—that is, if $(Au^*)_j > 2$—we define $\widetilde{A}_{j:} \stackrel{\text{def}}{=} \frac{2}{(Au^*)_j} \cdot A_{j:}$ to be the same j-th row $A_{j:}$, but scaled down by a factor of $\frac{2}{(Au^*)_j}$. It is clear from this definition that

$$A_{ji} \geq \widetilde{A}_{ji} \quad \text{for all} \quad (i, j) \in [n] \times [m] \quad \text{and} \quad (1 + \varepsilon)\mathbb{1} \leq \widetilde{A}u^* \leq 2\mathbb{1}.$$

Lemma 6.10 (distance adjustment)

$$\begin{aligned} f_\mu(\mathsf{x}_k) - f_\mu(u^*) &\leq \langle \mathbb{1} - A^T p(\mathsf{x}_k), \mathsf{x}_k - u^* \rangle + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^* \rangle + \varepsilon\mathsf{OPT} \\ &= \langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - u^* \rangle + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^* \rangle + \varepsilon\mathsf{OPT} \end{aligned}$$

At high level, ignoring the negligible term $\varepsilon\mathsf{OPT}$, Lemma 6.10 strengthens the classical bound due to the extra term of $\langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^* \rangle$. This extra term is always non-positive since $\widetilde{A} \leq A$ coordinate-wise, but may be very negative in certain cases.

Proof of Lemma 6.10

$$
\begin{aligned}
f_\mu(\mathsf{x}_k) - f_\mu(u^*) &= \mu \sum_{j=1}^{m} \left(e^{\frac{1}{\mu}(1-(A\mathsf{x}_k)_j)} - e^{\frac{1}{\mu}(1-(Au^*)_j)} \right) + \langle \mathbb{1}, \mathsf{x}_k - u^* \rangle \\
&\overset{①}{\le} \mu \sum_{j=1}^{m} \left(e^{\frac{1}{\mu}(1-(A\mathsf{x}_k)_j)} - e^{\frac{1}{\mu}(1-(\widetilde{A}u^*)_j)} \right) + \langle \mathbb{1}, \mathsf{x}_k - u^* \rangle + \mu \cdot m \cdot e^{-1/\mu} \\
&\overset{②}{\le} \sum_{j=1}^{m} e^{\frac{1}{\mu}(1-(A\mathsf{x}_k)_j)} \cdot \left((\widetilde{A}u^*)_j - (A\mathsf{x}_k)_j \right) + \langle \mathbb{1}, \mathsf{x}_k - u^* \rangle + \varepsilon \mathsf{OPT} \\
&= \sum_{j=1}^{m} p_j(\mathsf{x}_k) \cdot \left((\widetilde{A}u^*)_j - (A\mathsf{x}_k)_j \right) + \langle \mathbb{1}, \mathsf{x}_k - u^* \rangle + \varepsilon \mathsf{OPT} \\
&= \sum_{j=1}^{m} p_j(\mathsf{x}_k) \cdot \left((Au^*)_j - (A\mathsf{x}_k)_j \right) + \langle \mathbb{1}, \mathsf{x}_k - u^* \rangle \\
&\quad + \sum_{j=1}^{m} p_j(\mathsf{x}_k) \cdot \left((\widetilde{A}u^*)_j - (Au^*)_j \right) + \varepsilon \mathsf{OPT} \\
&= \langle -A^T p(\mathsf{x}_k), \mathsf{x}_k - u^* \rangle + \langle \mathbb{1}, \mathsf{x}_k - u^* \rangle + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^* \rangle + \varepsilon \mathsf{OPT}.
\end{aligned}
$$

Above, ① is because if $(Au^*)_j \neq (\widetilde{A}u^*)_j$ for some j, then it must satisfy that $(\widetilde{A}u^*)_j = 2$, and therefore $-e^{\frac{1}{\mu}(1-(Au^*)_j)} \le -e^{\frac{1}{\mu}(1-(\widetilde{A}u^*)_j)} + e^{-1/\mu}$. ② uses the convexity inequality of $e^b - e^a \le e^b \cdot (b-a)$, and the fact that $\mu m e^{-1/\mu} \ll \varepsilon \mathsf{OPT}$.

6.4 Step 2: gradient truncation

For analysis purpose, let us separate the indices $i \in [n]$ into large and small ones.

Definition 6.11 We make the following definitions.

- Let $B_k \overset{\text{def}}{=} \{i \in [n] : \nabla_i f_\mu(\mathsf{x}_k) < -\beta\}$ and $[n] \setminus B_k$ be the set of *large and small indices*.
- Let $\xi_k \in [-\beta, 1]^n$ be the *truncated gradient* so that $\xi_{k,i} = \mathbb{T}^{\mathsf{c}}(\nabla_i f_\mu(x_k))$ for each $i \in [n]$.
- Let $\eta_k \in (-\infty, 0]^n$ be the *large gradient* so that $\nabla f_\mu(\mathsf{x}_k) = \xi_k + \eta_k$. It is clear that

$$\eta_{k,i} = 0 \quad \text{for every} \quad i \notin B, \quad \text{and} \quad \eta_{k,i} = (1+\beta) - (A^T p(\mathsf{x}_k))_i \quad \text{for every} \quad i \in B.$$

- Let $\widetilde{\eta}_k \in (-\infty, \infty)^n$ be the *adjusted large gradient* so that

$$\widetilde{\eta}_{k,i} = 0 \quad \text{for every} \quad i \notin B, \quad \text{and} \quad \widetilde{\eta}_{k,i} = (1+\beta) - (\widetilde{A}^T p(\mathsf{x}_k))_i \quad \text{for every} \quad i \in B.$$

We denote by $\eta_k^{(i)} = (0, \ldots, 0, \eta_{k,i}, 0, \ldots, 0)$, the vector that is zero at all coordinates other than i, and similarly $\xi_k^{(i)} = (0, \ldots, \xi_{k,i}, \ldots, 0)$ and $\widetilde{\eta}_k^{(i)} = (0, \ldots, \widetilde{\eta}_{k,i}, \ldots, 0)$. We emphasize that $\eta_k^{(i)} \neq \eta_k$, $\widetilde{\eta}_k^{(i)} \neq \widetilde{\eta}_k$, and $\xi_k^{(i)} \neq \xi_k$.

The following key lemma is very analogous to (3.1) in the packing LP analysis.

Lemma 6.12 *(distance upper bound)*

$$f_\mu(\mathsf{x}_k) - f_\mu(u^*) \leq \tfrac{(1-\tau)}{\tau}(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)) + \mathbf{E}_i\Big[\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\rangle\Big] + \mathbf{E}_i\Big[\langle n\widetilde{\eta}_k^{(i)}, -u^*\rangle\Big] + \varepsilon\mathsf{OPT}.$$

Note that if one uses $\eta_k^{(i)}$ instead of $\widetilde{\eta}_k^{(i)}$, then Lemma 6.12 becomes trivial to prove just like (3.1). The reason we can have the stronger term $\widetilde{\eta}_k^{(i)}$ is precisely due to the distance adjustment Lemma 6.10.

Proof of Lemma 6.12 We derive the following sequence of inequalities:

$$\begin{aligned}
&\big(f_\mu(\mathsf{x}_k) - f_\mu(u^*)\big) - \varepsilon\mathsf{OPT} \\
&\overset{①}{\leq} \langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - u^*\rangle + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^*\rangle \\
&= \langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{z}_{k-1}\rangle + \langle \nabla f_\mu(\mathsf{x}_k), \mathsf{z}_{k-1} - u^*\rangle + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^*\rangle \\
&\overset{②}{=} \frac{(1-\tau)}{\tau}\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{y}_{k-1} - \mathsf{x}_k\rangle + \langle \nabla f_\mu(\mathsf{x}_k), \mathsf{z}_{k-1} - u^*\rangle \\
&\quad + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^*\rangle \\
&\overset{③}{\leq} \frac{(1-\tau)}{\tau}(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)) + \langle \nabla f_\mu(\mathsf{x}_k), \mathsf{z}_{k-1} - u^*\rangle \\
&\quad + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^*\rangle \\
&= \frac{(1-\tau)}{\tau}(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)) + \langle \xi_k + \eta_k, \mathsf{z}_{k-1} - u^*\rangle \\
&\quad + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k), u^*\rangle \\
&\overset{④}{\leq} \frac{(1-\tau)}{\tau}(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)) + \langle \xi_k, \mathsf{z}_{k-1} - u^*\rangle \\
&\quad + \langle \widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k) - \eta_k, u^*\rangle \\
&\overset{⑤}{\leq} \frac{(1-\tau)}{\tau}(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)) + \langle \xi_k, \mathsf{z}_{k-1} - u^*\rangle + \langle -\widetilde{\eta}_k, u^*\rangle \\
&= \frac{(1-\tau)}{\tau}(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)) + \mathbf{E}_i\Big[\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\rangle + \langle -n\widetilde{\eta}_k^{(i)}, u^*\rangle\Big].
\end{aligned}$$

Above, ① is due to Lemma 6.10. ② is because $\mathsf{x}_k = \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$, which implies that $\tau(\mathsf{x}_k - \mathsf{z}_{k-1}) = (1-\tau)(\mathsf{y}_{k-1} - \mathsf{x}_k)$. ③ is by the convexity of $f_\mu(\cdot)$. ④ is because $\langle \eta_k, \mathsf{z}_{k-1}\rangle \leq 0$, since $\eta_k \leq 0$ while $\mathsf{z}_{k-1} \geq 0$. ⑤ needs some careful justification: for every $i \notin B_k$, we have $(\widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k))_i - \eta_{k,i} \leq 0 - 0 = -\widetilde{\eta}_{k,i}$; for every $i \in B_k$, we have

$$(\widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k))_i - \eta_{k,i} = (\widetilde{A}^T p(\mathsf{x}_k) - A^T p(\mathsf{x}_k))_i - \big((1+\beta) - (A^T p(\mathsf{x}_k))_i\big)$$
$$= -\big((1+\beta) - (\widetilde{A}^T p(\mathsf{x}_k))_i\big) = -\widetilde{\eta}_{k,i},$$

where the two equalities follow from the definitions of $\eta_{k,i}$ and $\widetilde{\eta}_{k,i}$.

6.5 Step 3: mirror descent guarantee

Our update $\mathsf{z}_k^{(i)} \stackrel{\text{def}}{=} \operatorname{argmin}_{z \in \Delta_{\mathsf{simplex}}} \left\{ V_{\mathsf{z}_{k-1}}(z) + \langle (1+\gamma) n \alpha_k \xi_k^{(i)}, z \rangle \right\}$ is, by its definition, a mirror descent step [12]. We begin by explaining an attempt that is too weak for obtaining the $\varepsilon^{-1.5}$ convergence rate.

Using the classical theory, it is not hard to repeat the proof of Lemma 3.7—although changing the distance function from $\|\cdot\|_A^2$ to $V_x(y)$—and obtain that, as long as $\xi_{k,i}$ is in $[-1, +1]$ for each coordinate i, for every $u \in \Delta_{\mathsf{simplex}}$,

$$\mathbf{E}_i\left[\alpha_k \langle n \xi_k^{(i)}, \mathsf{z}_{k-1} - u \rangle\right] \le V_{\mathsf{z}_{k-1}}(u) - \mathbf{E}_i\left[V_{\mathsf{z}_k^{(i)}}(u)\right] + O(\alpha_k^2 n)\mathsf{OPT}.$$

This inequality only yields a slower ε^{-2} convergence rate, and ± 1 is also know as the *width parameter* from the multiplicative-weight-update language [7].

In our lemma below, we make use of the fact $\xi_{k,i}$ is in $[-\beta, +1] \subseteq [-1, +1]$. In essence, this allows us to replace the $O(\alpha_k^2 n)$ factor with a better $O(\alpha_k^2 \beta n)$ factor. We call it the *negative-width technique*.[12] Formally,

Lemma 6.13 (mirror descent) *Denoting by* $\gamma \stackrel{\text{def}}{=} 2\alpha_T n$, *we have*

$$\mathbf{E}_i\left[\alpha_k \langle n \xi_k^{(i)}, \mathsf{z}_{k-1} - u^* \rangle\right] \le V_{\mathsf{z}_{k-1}}\left(\frac{u^*}{1+\gamma}\right) - \mathbf{E}_i\left[V_{\mathsf{z}_k^{(i)}}\left(\frac{u^*}{1+\gamma}\right)\right] + 12\mathsf{OPT} \cdot \gamma \alpha_k \beta.$$

The proof is somewhat technical and included in "Appendix D.4".

6.6 Step 4: gradient descent guarantee

We show our gradient step never increases the objective for all choices of i. In addition, it decreases the objective by an amount proportional to the adjusted large gradient $\widetilde{\eta}_k^{(i)}$.

Lemma 6.14 (gradient descent) *For every* $i \in [n]$, *we have*

(a) $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge 0$, *and*
(b) $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge \frac{\mu\beta}{12} \cdot \langle -\widetilde{\eta}_k^{(i)}, u^* \rangle$.

[12] This negative width technique is related to [7, Definition 3.2], where the authors analyze the multiplicative weight update method in a special case when the oracle returns loss values only in $[-\ell, +\rho]$, for some $\ell \ll \rho$. This technique is also a sub-case of a more general theory of mirror descent, known as the local-norm convergence, that we have summarized in a separate and later paper [3].

The proof of Lemma 6.14 is quite technical and can be found in "Appendix D.4".

At high level, one would hope to prove that the gradient step decreases the objective by an amount proportional to the large gradient $\eta_k^{(i)}$, rather than the adjusted large gradient $\widetilde{\eta}_k^{(i)}$. If that were true, the entire proof structure of our covering LP convergence would become much closer to that of packing LP, and there would be absolutely no need for the introduction of the distance adjustment in Sect. 6.3, as well as the definitions of $\widetilde{A}$ and $\widetilde{\eta}$.

Unfortunately, if one replaces $\widetilde{\eta}$ with η in the above lemma, the inequality is *false*. The reason behind it is very similar to what we have summarized in Sect. 6.3, and related to the fact that the exponential penalty function is not Lipschitz smooth.

6.7 Step 5: putting all together

Combining Lemma 6.12, Lemma 6.13, and Lemma 6.14, we obtain that

$$\begin{aligned}
&\alpha_k\big(f_\mu(\mathsf{x}_k) - f_\mu(u^*)\big) - \alpha_k\varepsilon\mathsf{OPT} \\
&\quad\le \frac{(1-\tau)\alpha_k}{\tau}\big(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)\big) + \mathbf{E}_i\Big[\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\rangle\Big] \\
&\qquad + \mathbf{E}_i\Big[\alpha_k\langle n\widetilde{\eta}_k^{(i)}, -u^*\rangle\Big] \\
&\quad\le \frac{(1-\tau)\alpha_k}{\tau}\big(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)\big) + V_{\mathsf{z}_{k-1}}\Big(\frac{u^*}{1+\gamma}\Big) - \mathbf{E}_i\Big[V_{\mathsf{z}_k^{(i)}}\Big(\frac{u^*}{1+\gamma}\Big)\Big] \\
&\qquad + 12\mathsf{OPT}\cdot\gamma\alpha_k\beta + \mathbf{E}_i\Big[\frac{12\alpha_k n}{\mu\beta}\big(f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})\big)\Big]
\end{aligned}$$

Remark 6.15 Above, the quantity "$12\mathsf{OPT}\cdot\gamma\alpha_k\beta$" is the loss term introduced by the mirror descent. Unlike the packing LP case—see (3.2)—this loss term is not dominated by the gradient step. (If one could do so, this would give *CovLPSolver* an ε^{-1} convergence rate.)

The quantity "$\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\rangle$" is the loss introduced by the (adjusted) large gradient $\widetilde{\eta}$, and is dominated by our gradient step progress owing to Lemma 6.14. This is similar to the packing LP case—see Lemma 3.10.

From here, let us use the special choice of $\tau = \frac{\mu\beta}{12n}$. We obtain that

$$\begin{aligned}
&-\alpha_k\big(f_\mu(u^*) + \varepsilon\mathsf{OPT}\big) \\
&\quad\le 12\gamma\alpha_k\beta\mathsf{OPT} + \frac{(1-\tau)\alpha_k}{\tau}f_\mu(\mathsf{y}_{k-1}) + V_{\mathsf{z}_{k-1}}\Big(\frac{u^*}{1+\gamma}\Big) \\
&\qquad - \mathbf{E}_i\Big[\frac{\alpha_k}{\tau}f_\mu(\mathsf{y}_k^{(i)}) + V_{\mathsf{z}_k^{(i)}}\Big(\frac{u^*}{1+\gamma}\Big)\Big].
\end{aligned}$$

Using the choice $\alpha_k = \frac{\alpha_{k-1}}{1-\tau}$ and telescoping the above inequality for $k = 1, \ldots, T$, we have

$$-\Big(\sum_{k=1}^{T}\alpha_k\Big)\big(f_\mu(u^*)+\varepsilon\mathsf{OPT}\big) \le \Big(\sum_{k=1}^{T}\alpha_k\Big)\cdot 12\gamma\beta\mathsf{OPT} + \frac{\alpha_0}{\tau}f_\mu(\mathsf{y}_0) + V_{\mathsf{z}_0}\Big(\frac{u^*}{1+\gamma}\Big) - \frac{\alpha_T}{\tau}\mathbf{E}\big[f_\mu(\mathsf{y}_T)\big].$$

We compute that $\sum_{k=1}^{T}\alpha_k = \alpha_T\cdot\sum_{k=0}^{T-1}(1-\tau)^k = \alpha_T\cdot\frac{1-(1-\tau)^T}{\tau} < \frac{\alpha_T}{\tau}$, and recall that $\gamma = 2\alpha_T n$. Therefore, we rearrange and get

$$\begin{aligned}\frac{\alpha_T}{\tau}\mathbf{E}\big[f_\mu(\mathsf{y}_T)\big] &\le \frac{\alpha_T}{\tau}\big(f_\mu(u^*)+\varepsilon\mathsf{OPT}\big) + \frac{\alpha_T}{\tau}\cdot 12\gamma\beta\mathsf{OPT} + \frac{\alpha_0}{\tau}f_\mu(\mathsf{y}_0) \\ &\quad + V_{\mathsf{z}_0}\Big(\frac{u^*}{1+\gamma}\Big), \\ \implies \mathbf{E}\big[f_\mu(\mathsf{y}_T)\big] &\le f_\mu(u^*) + \varepsilon\mathsf{OPT} + 24\alpha_T n\beta\mathsf{OPT} + (1-\tau)^T f_\mu(\mathsf{y}_0) \\ &\quad + \frac{\tau}{\alpha_T}V_{\mathsf{z}_0}\Big(\frac{u^*}{1+\gamma}\Big). \end{aligned} \tag{6.3}$$

From this point, we need to use our special choice of the initial point $\mathsf{x}_0 = \mathsf{y}_0 = \mathsf{z}_0 = x^{\mathsf{start}}$ (see Proposition 6.2), which implies that $f_\mu(\mathsf{y}_0) \le 4\mathsf{OPT}$ and $\mathbb{1}^T x^{\mathsf{start}} \le 4\mathsf{OPT}$. We also have

$$\begin{aligned}V_{\mathsf{z}_0}\Big(\frac{u^*}{1+\gamma}\Big) = V_{x^{\mathsf{start}}}\Big(\frac{u^*}{1+\gamma}\Big) &= \sum_{i=1}^{n}\frac{u_i^*}{1+\gamma}\log\frac{u_i^*}{(1+\gamma)x_i^{\mathsf{start}}} + x_i^{\mathsf{start}} - \frac{u_i^*}{1+\gamma} \\ &\overset{①}{\le} \sum_{i=1}^{n} u_i^*\log(u_i^*\cdot n) + 4\mathsf{OPT} \overset{②}{\le} (2\log(nm)+4)\cdot\mathsf{OPT}.\end{aligned}$$

Above, inequality ① follows because $x_i^{\mathsf{start}} \ge 1/n$ for all $i \in [n]$ according to the definition in Proposition 6.2; inequality ② follows because each $u_i^* \le (1+\varepsilon/2)x_i^* \le (1+\varepsilon/2)\mathsf{OPT} \le (1+\varepsilon/2)m$ and $\mathbb{1}^T u_i^* = (1+\varepsilon/2)\mathsf{OPT}$, as well as the fact that ε is sufficiently small.

Finally, we choose $\beta = \sqrt{\varepsilon}$, $T = \lceil\frac{1}{\tau}\log(1/\varepsilon)\rceil$, and α_0 such that $\alpha_T = \frac{\varepsilon}{12n\beta}$. Substituting into (6.3) all of these parameters, along with the aforementioned inequalities $f_\mu(\mathsf{y}_0) \le 4\mathsf{OPT}$ and $V_{\mathsf{z}_0}\big(\frac{u^*}{1+\gamma}\big) \le (2\log(nm)+4)\cdot\mathsf{OPT}$, as well as $f_\mu(u^*) \le (1+\varepsilon)\mathsf{OPT}$ from Proposition 4.5.a, we obtain that

$$\begin{aligned}\mathbf{E}\big[f_\mu(\mathsf{y}_T)\big] &\le (1+\varepsilon)\mathsf{OPT} + \varepsilon\mathsf{OPT} + 2\varepsilon\mathsf{OPT} + \varepsilon f_\mu(\mathsf{y}_0) \\ &\quad + \frac{\mu\beta/12n}{\varepsilon/12n\beta}(2\log(nm)+4)\mathsf{OPT} \\ &\le (1+9\varepsilon)\mathsf{OPT}.\end{aligned}$$

This finishes the proof of Theorem 6.6. □

Appendix

A.1 Proof of Lemma 3.6

Lemma 3.6 *We have* $\mathsf{x}_k, \mathsf{y}_k, \mathsf{z}_k \in \Delta_{\mathsf{box}}$ *for all* $k = 0, 1, \ldots, T$.

Proof This is true at the beginning as $\mathsf{x}_0 = \mathsf{y}_0 = x^{\mathsf{start}} \in \Delta_{\mathsf{box}}$ (see Fact 2.8) and $\mathsf{z}_0 = 0 \in \Delta_{\mathsf{box}}$. In fact, it suffices for us to show that for every $k \geq 1$, $\mathsf{y}_k = \sum_{l=0}^{k} \gamma_k^l \mathsf{z}_l$ for some scalars γ_k^l satisfying $\sum_l \gamma_k^l = 1$ and $\gamma_k^l \geq 0$ for each $l = 0, \ldots, k$. If this is true, we can prove the lemma by induction: at each iteration $k \geq 1$,

1. $\mathsf{x}_k = \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$ must be in Δ_{box} because y_{k-1} and z_{k-1} are and $\tau \in [0, 1]$,
2. z_k is in Δ_{box} by the definition that $\mathsf{z}_k = \operatorname{argmin}_{z \in \Delta_{\mathsf{box}}}\{\cdots\}$, and
3. y_k is also in Δ_{box} because $\mathsf{y}_k = \sum_{l=0}^{k} \gamma_k^l \mathsf{z}_l$ is a convex combination of the z_l's and Δ_{box} is convex.

For the rest of the proof, we show that $\mathsf{y}_k = \sum_{l=0}^{k} \gamma_k^l \mathsf{z}_l$ for every $k \geq 1$ with coefficients [13]

$$\gamma_k^l = \begin{cases} (1-\tau)\gamma_{k-1}^l, & l = 0, \ldots, k-2; \\ \left(\frac{1}{n\alpha_{k-1}L} - \frac{1}{n\alpha_k L}\right) + \tau\left(1 - \frac{1}{n\alpha_{k-1}L}\right), & l = k-1; \\ \frac{1}{n\alpha_k L}, & l = k. \end{cases}$$

This is true at the base case $k = 1$ because $\mathsf{y}_1 = \mathsf{x}_1 + \frac{1}{n\alpha_1 L}(\mathsf{z}_1 - \mathsf{z}_0) = \frac{1}{n\alpha_1 L}\mathsf{z}_1 + \left(1 - \frac{1}{n\alpha_1 L}\right)\mathsf{z}_0$. For the general $k \geq 2$, we have

$$\begin{aligned} \mathsf{y}_k &= \mathsf{x}_k + \frac{1}{n\alpha_k L}(\mathsf{z}_k - \mathsf{z}_{k-1}) \\ &= \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1} + \frac{1}{n\alpha_k L}(\mathsf{z}_k - \mathsf{z}_{k-1}) \\ &= \tau \mathsf{z}_{k-1} + (1-\tau)\left(\sum_{l=0}^{k-2} \gamma_{k-1}^l \mathsf{z}_l + \frac{1}{n\alpha_{k-1}L}\mathsf{z}_{k-1}\right) + \frac{1}{n\alpha_k L}(\mathsf{z}_k - \mathsf{z}_{k-1}) \\ &= \left(\sum_{l=0}^{k-2}(1-\tau)\gamma_{k-1}^l \mathsf{z}_l\right) + \left(\left(\frac{1}{n\alpha_{k-1}L} - \frac{1}{n\alpha_k L}\right) + \tau\left(1 - \frac{1}{n\alpha_{k-1}L}\right)\right)\mathsf{z}_{k-1} \\ &\quad + \frac{1}{n\alpha_k L}\mathsf{z}_k. \end{aligned}$$

Therefore, we obtain $\mathsf{y}_k = \sum_{l=0}^{k} \gamma_k^l \mathsf{z}_l$ as desired.

[13] We wish to point out that this proof coincides with a lemma from the accelerated coordinate descent theory of Fercoq and Richtárik [17]. Their paper is about optimizing an objective function that is Lipschitz smooth, and thus irrelevant to our work.

It is now easy to check that under our definition of α_k (which satisfies $\alpha_k \geq \alpha_{k-1}$ and $\alpha_k \geq \alpha_0 = \frac{1}{nL}$, we must have $\gamma_k^l \geq 0$ for all k and l. Also,

$$\begin{aligned}\sum_l \gamma_k^l &= \sum_{l=0}^{k-2}(1-\tau)\gamma_{k-1}^l + \left(\left(\frac{1}{n\alpha_{k-1}L} - \frac{1}{n\alpha_k L}\right) + \tau\left(1 - \frac{1}{n\alpha_{k-1}L}\right)\right) + \frac{1}{n\alpha_k L} \\ &= (1-\tau)\left(1 - \frac{1}{n\alpha_{k-1}L}\right) + \left(\left(\frac{1}{n\alpha_{k-1}L} - \frac{1}{n\alpha_k L}\right) + \tau\left(1 - \frac{1}{n\alpha_{k-1}L}\right)\right) \\ &\quad + \frac{1}{n\alpha_k L} = 1.\end{aligned}$$

□

B.2 Proof of Proposition 4.5

Proposition 4.5 (a) $f_\mu(u^*) \leq (1+\varepsilon)\mathsf{OPT}$ *for* $u^* \stackrel{\text{def}}{=} (1+\varepsilon/2)x^*$.
(b) $f_\mu(x) \geq (1-\varepsilon)\mathsf{OPT}$ *for every* $x \geq 0$.
(c) *For any* $x \geq 0$ *satisfying* $f_\mu(x) \leq 2\mathsf{OPT}$, *we must have* $Ax \geq (1-\varepsilon)\mathbb{1}$.
(d) *If* $x \geq 0$ *satisfies* $f_\mu(x) \leq (1+\delta)\mathsf{OPT}$ *for some* $\delta \in [0,1]$, *then* $\frac{1}{1-\varepsilon}x$ *is a* $\frac{1+\delta}{1-\varepsilon}$*-approximate solution to the covering LP.*

Proof (a) We have $\mathbb{1}^T u^* = (1+\varepsilon/2)\mathsf{OPT}$ by the definition of OPT. Also, from the feasibility constraint $Ax^* \geq \mathbb{1}$ in the covering LP, we have $Au^* - \mathbb{1} \geq \varepsilon/2 \cdot \mathbb{1}$, and can compute $f_\mu(u^*)$ as follows:

$$\begin{aligned}f_\mu(u^*) = \mu\sum_j e^{\frac{1}{\mu}(1-(Au^*)_j)} + \mathbb{1}^T u^* &\leq \mu\sum_j e^{\frac{-\varepsilon/2}{\mu}} + (1+\varepsilon/2)\mathsf{OPT} \\ &\leq \frac{\mu m}{(nm)^2} + (1+\varepsilon/2)\mathsf{OPT} \leq (1+\varepsilon)\mathsf{OPT}.\end{aligned}$$

(b) Suppose towards contradiction that $f_\mu(x) < (1-\varepsilon)\mathsf{OPT}$. Since $f_\mu(x) < \mathsf{OPT} \leq m$, we must have that for every $j \in [m]$, it satisfies that $e^{\frac{1}{\mu}(1-(Ax)_j)} \leq f_\mu(x)/\mu \leq m/\mu$. This further implies $(Ax)_j \geq 1-\varepsilon$ by the definition of μ. In other words, $Ax \geq (1-\varepsilon)\mathbb{1}$. By the definition of OPT, we must then have $\mathbb{1}^T x \geq (1-\varepsilon)\mathsf{OPT}$, finishing the proof that $f_\mu(x) \geq \mathbb{1}^T x \geq (1-\varepsilon)\mathsf{OPT}$, giving a contradiction.
(c) To show $Ax \geq (1-\varepsilon)\mathbb{1}$, we can assume that $v = \max_j(1-(Ax)_j) > \varepsilon$ because otherwise we are done. Under this definition, we have

$$f_\mu(x) \geq \mu e^{\frac{v}{\mu}} = \mu\left((\tfrac{nm}{\varepsilon})^4\right)^{v/\varepsilon} \geq \frac{\varepsilon}{4\log(nm/\varepsilon)}(\tfrac{nm}{\varepsilon})^4 \gg 2\mathsf{OPT},$$

contradicting to our assumption that $f_\mu(x) \leq 2\mathsf{OPT}$. Therefore, we must have $v \leq \varepsilon$, that is, $Ax \geq (1-\varepsilon)\mathbb{1}$.
(d) For any x satisfying $f_\mu(x) \leq (1+\theta)\mathsf{OPT} \leq 2\mathsf{OPT}$, owing to Proposition 4.5c, we first have that x is approximately feasible, i.e., $Ax \geq (1-\varepsilon)\mathbb{1}$. Next, because

$\mathbb{1}^T x \le f_\mu(x) \le (1+\theta)\mathsf{OPT}$, we know that x yields an objective $\mathbb{1}^T x \le (1+\theta)\mathsf{OPT}$. Letting $x' = \frac{1}{1-\varepsilon}x$, we both have that x' is feasible (i.e., $Ax' \ge \mathbb{1}$), and x' has an objective $\mathbb{1}^T x'$ at most $\frac{1+\delta}{1-\varepsilon}\mathsf{OPT}$.

□

C.3 Missing proofs for Sect. 5

In this section we prove Theorem 5.3. Because the proof structure is almost identical to that of Theorem 3.4, we spend most of the discussions only pointing out the difference rather than repeating the proofs. The following three lemmas are completely identical to the ones in the packing LP case, so we restate them below:

Lemma C.1 (cf. Lemma 3.3) *Each iteration of* $CovLPSolver^{\mathsf{wb}}$ *can be implemented to run in expected* $O(N/n)$ *time.*

Lemma C.2 (cf. Lemma 3.6) *We have* $\mathsf{x}_k, \mathsf{y}_k, \mathsf{z}_k \in \Delta_{\mathsf{box}}$ *for all* $k = 0, 1, \ldots, T$.

Lemma C.3 (cf. Lemma 3.7) *For every* $u \in \Delta_{\mathsf{box}}$, *it satisfies* $\langle n\alpha_k \xi_k^{(i)}, \mathsf{z}_{k-1} - u\rangle \le n^2\alpha_k^2 L \cdot \langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)}\rangle + \frac{1}{2}\|\mathsf{z}_{k-1} - u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k^{(i)} - u\|_A^2$.

For the gradient descent guarantee of Sect. 3.3, one can first note that Lemma 2.7 remains true: this can be verified by replacing $\nabla_i f_\mu(x)+1$ in its proof with $1-\nabla_i f_\mu(x)$. For this reason, Lemma 3.9 (which is built on Lemma 2.7) also remains true. We state it below:

Lemma C.4 (cf. Lemma 3.9) *We have* $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge \frac{1}{2}\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)}\rangle \ge 0$.

Putting all together Denote by $\eta_k^{(i)} \in \mathbb{R}^n_{\le 0}$ the vector that is only non-zero at coordinate i, and satisfies $\eta_{k,i}^{(i)} = \nabla_i f_\mu(\mathsf{x}_k) - \xi_{k,i}^{(i)} \in (-\infty, 0]$. In other words, the full gradient

$$\nabla f_\mu(\mathsf{x}_k) = \mathbf{E}_i[(0, \ldots, n\nabla_i f_\mu(\mathsf{x}_k), \ldots, 0)] = \mathbf{E}_i[n\eta_k^{(i)} + n\xi_k^{(i)}]$$

can be (in expectation) decomposed into the a large but non-positive component $\eta_k^{(i)} \in (-\infty, 0]^n$ and a small component $\xi_k^{(i)} \in [-1, 1]^n$. Similar as Sect. 3.4, for any $u \in \Delta_{\mathsf{box}}$, we can use a basic convexity argument and the mirror descent lemma to compute that

$$\begin{aligned}
\alpha_k(f_\mu(\mathsf{x}_k) - f_\mu(u)) &\le \langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - u\rangle \\
&= \langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{z}_{k-1}\rangle + \langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{z}_{k-1} - u\rangle \\
&= \langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{z}_{k-1}\rangle + \mathbf{E}_i\left[\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u\rangle + \langle n\alpha_k \xi_k^{(i)}, \mathsf{z}_{k-1} - u\rangle\right]
\end{aligned}$$

$$\begin{aligned}
&\overset{①}{=} \frac{(1-\tau)\alpha_k}{\tau}\left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{y}_{k-1} - \mathsf{x}_k \right\rangle \\
&\quad + \mathbf{E}_i\left[\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u\right\rangle + \left\langle n\alpha_k \xi_k^{(i)}, \mathsf{z}_{k-1} - u\right\rangle\right] \qquad \text{(C.1)} \\
&\overset{②}{\le} \frac{(1-\tau)\alpha_k}{\tau}\left(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)\right) \\
&\quad + \mathbf{E}_i\Bigg[\boxed{\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u\right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle} \\
&\quad + \frac{1}{2}\|\mathsf{z}_{k-1} - u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k^{(i)} - u\|_A^2\Bigg] \qquad \text{(C.2)}
\end{aligned}$$

Above, ① is because $\mathsf{x}_k = \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$, which implies that $\tau(\mathsf{x}_k - \mathsf{z}_{k-1}) = (1-\tau)(\mathsf{y}_{k-1} - \mathsf{x}_k)$. ② uses convexity and Lemma C.3. We can establish the following lemma to upper bound the boxed term in (C.2). Its proof is in the same spirit to that of Lemma 3.10, and is the *only* place that we require all vectors to reside in Δ_{box}.

Lemma C.5 (cf. Lemma 3.10) *For every $u \in \Delta_{\mathsf{box}}$,*

$$\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u\right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle \le 21 n\alpha_k L \cdot \left(f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})\right).$$

Proof of Lemma C.5 Now there are three possibilities:

– If $\eta_{k,i}^{(i)} = 0$, then we must have $\xi_{k,i}^{(i)} = \nabla_i f_\mu(\mathsf{x}_k) \in [-1, 1]$. Lemma C.4 implies

$$\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u\right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle \\
&\quad = n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle \le 2n^2\alpha_k^2 L \cdot \left(f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})\right)
\end{aligned}$$

– If $\eta_{k,i}^{(i)} < 0$ and $\mathsf{z}_{k,i}^{(i)} < \frac{10}{\|A_{:i}\|_\infty}$ (thus $\mathsf{z}_k^{(i)}$ is not on the boundary of Δ_{box}), then we precisely have $\mathsf{z}_{k,i}^{(i)} = \mathsf{z}_{k-1,i} + \frac{n\alpha_k}{\|A_{:i}\|_\infty}$, and accordingly $\mathsf{y}_{k,i}^{(i)} = \mathsf{x}_{k,i} + \frac{1}{L\|A_{:i}\|_\infty} > \mathsf{x}_{k,i}$. In this case,

$$\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u\right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle \\
&\overset{①}{\le} n\alpha_k \cdot \nabla_i f_\mu(\mathsf{x}_k) \cdot \frac{-10}{\|A_{:i}\|_\infty} + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle \\
&\overset{②}{<} n\alpha_k \cdot \nabla_i f_\mu(\mathsf{x}_k) \cdot \frac{-10}{\|A_{:i}\|_\infty} + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle \\
&\overset{③}{=} 10 n\alpha_k L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)}\right\rangle \\
&\overset{④}{\le} \left(20 n\alpha_k L + 2n^2\alpha_k^2 L\right) \cdot \left(f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})\right).
\end{aligned}$$

Above, ① follows from the fact that $\mathsf{z}_{k-1}, u \in \Delta_{\mathsf{box}}$ and therefore $z_{k-1,i} \ge 0$ and $u_i \le \frac{10}{\|A_{:i}\|_\infty}$ by the definition of Δ_{box}, and $u \ge 0$; ② follows from the fact that x_k and $\mathsf{y}_k^{(i)}$ are only different at coordinate i, and $\xi_{k,i}^{(i)} = -1 > \nabla_i f_\mu(\mathsf{x}_k)$ (since $\eta_{k,i}^{(i)} < 0$); ③ follows from the fact that $\mathsf{y}_k^{(i)} = \mathsf{x}_k + \frac{\mathbf{e}_i}{L\|A_{:i}\|_\infty}$; and ④ uses Lemma C.4.

– If $\eta_{k,i}^{(i)} < 0$ and $\mathsf{z}_{k,i}^{(i)} = \frac{10}{\|A_{:i}\|_\infty}$, then we have

$$\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{①}{\le} \left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)} \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{②}{\le} \left\langle n\alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)} \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{③}{=} n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\overset{④}{\le} 4n^2\alpha_k^2 L \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})).
\end{aligned}$$

Above, ① is because $u_i \le \frac{10}{\|A_{:i}\|_\infty} = \mathsf{z}_{k,i}^{(i)}$ and $\eta_{k,i}^{(i)} < 0$, together with $\nabla_i f_\mu(\mathsf{x}_k) < \xi_{k,i}^{(i)}$ and $\mathsf{x}_{k,i} \le \mathsf{y}_{k,i}^{(i)}$; ② uses $\nabla_i f_\mu(\mathsf{x}_k) = \eta_{k,i}^{(i)} - 1 < \eta_{k,i}^{(i)}$ and $\mathsf{z}_{k,i}^{(i)} \ge \mathsf{z}_{k-1,i}$; ③ is from our choice of y_k which satisfies that $\mathsf{z}_{k-1} - \mathsf{z}_k^{(i)} = n\alpha_k L(\mathsf{x}_k - \mathsf{y}_k^{(i)})$; and ④ uses Lemma C.4.

Combining the three cases, and using the fact that $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge 0$, we conclude that

$$\begin{aligned}
&\left\langle n\alpha_k \eta_k^{(i)}, \mathsf{z}_{k-1} - u \right\rangle + n^2\alpha_k^2 L \cdot \left\langle \xi_k^{(i)}, \mathsf{x}_k - \mathsf{y}_k^{(i)} \right\rangle \\
&\le (20n\alpha_k L + 4n^2\alpha_k^2 L) \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})) \\
&\le 21n\alpha_k L \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})).
\end{aligned}$$

Above, the last inequality uses our choice of α_k, which implies $n\alpha_k \le n\alpha_T = \frac{1}{\varepsilon L} \le \frac{1}{4}$.

Plugging Lemma C.5 back to (C.2), we have

$$\begin{aligned}
&\alpha_k(f_\mu(\mathsf{x}_k) - f_\mu(u)) \le \left\langle \alpha_k \nabla f_\mu(\mathsf{x}_k), \mathsf{x}_k - u \right\rangle \\
&\overset{①}{\le} \frac{(1-\tau)\alpha_k}{\tau}(f_\mu(\mathsf{y}_{k-1}) - f_\mu(\mathsf{x}_k)) \\
&\quad + \mathbf{E}_i\left[21n\alpha_k L \cdot (f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)})) + \frac{1}{2}\|\mathsf{z}_{k-1} - u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k - u\|_A^2\right] \\
&\overset{②}{\le} \alpha_k f_\mu(\mathsf{x}_k) + \left(21n\alpha_k L - \alpha_k\right) f_\mu(\mathsf{y}_{k-1}) \\
&\quad + \mathbf{E}_i\left[-21n\alpha_k L \cdot f_\mu(\mathsf{y}_k^{(i)}) + \frac{1}{2}\|\mathsf{z}_{k-1} - u\|_A^2 - \frac{1}{2}\|\mathsf{z}_k - u\|_A^2\right]. \qquad \text{(C.3)}
\end{aligned}$$

Above, ① uses Lemma C.5; and ② is because we have chosen τ to satisfy $\frac{1}{\tau} = 21nL$.

Next, recall that we have picked α_k so that $(21nL - 1)\alpha_k = 21nL \cdot \alpha_{k-1}$ in $CovLPSolver^{\mathsf{wb}}$. Telescoping (C.3) for $k = 1, \ldots, T$ and choosing $u^* = (1 + \varepsilon/2)x^*$, we have

$$-\sum_{k=1}^{T}\alpha_k f_\mu(u^*) \le 21 f_\mu(\mathsf{y}_0) - 21n\alpha_T L$$
$$\cdot\mathbf{E}[f_\mu(\mathsf{y}_T)] + \|\mathsf{z}_0 - u^*\|_A^2 \le -21n\alpha_T L \cdot \mathbf{E}[f_\mu(\mathsf{y}_T)] + 75\mathsf{OPT}.$$

Here, the second inequality is due to $f_\mu(\mathsf{y}_0) = f_\mu(x^{\mathsf{start}}) \le 3\mathsf{OPT}$ from Fact 5.2, and the fact that

$$\|\mathsf{z}_0 - u^*\|_A^2 = \|u^*\|_A^2 = \sum_{i=1}^{n}(u_i^*)^2 \cdot \|A_{:i}\|_\infty \le (1+\varepsilon/2)^2 \sum_{i=1}^{n}(x_i^*)^2 \cdot \|A_{:i}\|_\infty$$
$$\le 10(1+\varepsilon/2)^2 \sum_{i=1}^{n} x_i^* < 12\mathsf{OPT}.$$

Finally, using the fact that $\sum_{k=1}^{T}\alpha_k = \alpha_T \cdot \sum_{k=0}^{T-1}\left(1-\frac{1}{21nL}\right)^k = 21n\alpha_T L\left(1-(1-\frac{1}{21nL})^T\right)$, we rearrange and obtain that

$$\mathbf{E}[f_\mu(\mathsf{y}_T)] \le \frac{\sum_k \alpha_k}{21n\alpha_T L} f_\mu(u^*) + \frac{75}{21n\alpha_T L}\mathsf{OPT} = \left(1-(1-\frac{1}{21nL})^T\right) f_\mu(u^*)$$
$$+ \frac{75}{21n\alpha_T L}\mathsf{OPT}.$$

We choose $T = \lceil 21nL\log(1/\varepsilon)\rceil$ so that $\frac{1}{n\alpha_T L} = (1-\frac{1}{21nL})^T \le \varepsilon$. Combining this with the fact that $f_\mu(u^*) \le (1+\varepsilon)\mathsf{OPT}$ (see Proposition 4.5a), we obtain

$$\mathbf{E}[f_\mu(\mathsf{y}_T)] \le (1+\varepsilon)\mathsf{OPT} + 3.6\varepsilon\cdot\mathsf{OPT} < (1+4.6\varepsilon)\mathsf{OPT}.$$

Therefore, we have finished proving Theorem 5.3. □

D.4 Missing proofs for Sect. 6

Proposition 6.4 *If $\mathsf{z}_{k-1} \in \Delta_{\mathsf{simplex}}$ and $\mathsf{z}_{k-1} > 0$, the minimizer $z = \operatorname{argmin}_{z\in\Delta_{\mathsf{simplex}}}\left\{V_{\mathsf{z}_{k-1}}(z) + \langle\delta\mathbf{e}_i, z\rangle\right\}$ for any scalar $\delta \in \mathbb{R}$ and basis vector $\mathbf{e}_i$ can be computed as follows:*

1. $z \leftarrow \mathsf{z}_{k-1}$.
2. $z_i \leftarrow z_i \cdot e^{-\delta}$.
3. *If* $\mathbb{1}^T z > 2\mathsf{OPT}'$, $z \leftarrow \frac{2\mathsf{OPT}'}{\mathbb{1}^T z} z$.
4. *Return* z.

Proof Let us denote by z the returned value of the described procedure, and $g(u) \stackrel{\text{def}}{=} V_{\mathsf{z}_{k-1}}(u) + \langle\delta\mathbf{e}_i, u\rangle$. Since $\Delta_{\mathsf{simplex}}$ is a convex body and $g(\cdot)$ is convex, to show $z = \operatorname{argmin}_{z\in\Delta_{\mathsf{simplex}}}\{g(u)\}$, it suffices for us to prove that for every $u \in \Delta_{\mathsf{simplex}}$,

$\langle \nabla g(z), u-z\rangle \geq 0$. Since the gradient $\nabla g(z)$ can be written explicitly, this is equivalent to

$$\delta(u_i - z_i) + \sum_{\ell=1}^n \log \tfrac{z_\ell}{\mathsf{z}_{k-1,\ell}} \cdot (u_\ell - z_\ell) \geq 0.$$

If the re-scaling in step 3 is not executed, then we have $z_\ell = \mathsf{z}_{k-1,\ell}$ for every $\ell \neq i$, and $z_i = \mathsf{z}_{k-1,i} \cdot e^{-\delta}$; thus, the left-hand side is zero so the above inequality is true for every $u \in \Delta_{\mathsf{simplex}}$.

Otherwise, we have $\mathbb{1}^T z = 2\mathsf{OPT}'$ and there exists some constant $Z > 1$ such that, $z_\ell = \mathsf{z}_{k-1,\ell}/Z$ for every $\ell \neq i$, and $z_i = \mathsf{z}_{k-1,i} \cdot e^{-\delta}/Z$. In such a case, the left-hand side equals to

$$(u_i - z_i) \cdot (\delta - \delta) + \sum_{\ell=1}^n - \log Z \cdot (u_\ell - z_\ell).$$

It is clear at this moment that since $\log Z > 0$ and $\mathbb{1}^T u \leq 2\mathsf{OPT}' = \mathbb{1}^T z$, the above quantity is always non-negative, finishing the proof. □

Lemma 6.13 *Denoting by* $\gamma \stackrel{\text{def}}{=} 2\alpha_T n$, *we have*

$$\mathbf{E}_i\big[\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\rangle\big] \leq V_{\mathsf{z}_{k-1}}\big(\frac{u^*}{1+\gamma}\big) - \mathbf{E}_i\Big[V_{\mathsf{z}_k^{(i)}}\big(\frac{u^*}{1+\gamma}\big)\Big] + 12\mathsf{OPT}\cdot\gamma\alpha_k\beta.$$

Proof Define $w(x) \stackrel{\text{def}}{=} \sum_i x_i \log(x_i) - x_i$ and accordingly, $V_x(y) = w(y) - \langle \nabla w(x), y-x\rangle - w(x) = \sum_i y_i \log \frac{y_i}{x_i} + x_i - y_i$. We first compute using the classical analysis of mirror descent step as follows:

$$\begin{aligned}
&\gamma\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1}\rangle + \alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\rangle \\
&= (1+\gamma)\alpha_k\Big\langle n\xi_k^{(i)}, \mathsf{z}_k^{(i)} - \frac{u^*}{1+\gamma}\Big\rangle + (1+\gamma)\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)}\rangle \\
&\stackrel{①}{\leq} \Big\langle \nabla w(\mathsf{z}_{k-1}) - \nabla w(\mathsf{z}_k^{(i)}), \mathsf{z}_k^{(i)} - \frac{u^*}{1+\gamma}\Big\rangle + (1+\gamma)\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)}\rangle \\
&= \Big(w\big(\frac{u^*}{1+\gamma}\big) - w(\mathsf{z}_{k-1}) - \Big\langle \nabla w(\mathsf{z}_{k-1}), \frac{u^*}{1+\gamma} - \mathsf{z}_{k-1}\Big\rangle\Big) \\
&\quad - \Big(w\big(\frac{u^*}{1+\gamma}\big) - w(\mathsf{z}_k^{(i)}) - \Big\langle \nabla w(\mathsf{z}_k^{(i)}), \frac{u^*}{1+\gamma} - \mathsf{z}_k^{(i)}\Big\rangle\Big) \\
&\quad + \Big(w(\mathsf{z}_{k-1}) - w(\mathsf{z}_k^{(i)}) - \big\langle \nabla w(\mathsf{z}_{k-1}), \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)}\big\rangle\Big) \\
&\quad + (1+\gamma)\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)}\rangle \\
&= V_{\mathsf{z}_{k-1}}\big(\frac{u^*}{1+\gamma}\big) - V_{\mathsf{z}_k^{(i)}}\big(\frac{u^*}{1+\gamma}\big) + \boxed{(1+\gamma)\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)}\rangle - V_{\mathsf{z}_{k-1}}(\mathsf{z}_k^{(i)})}.
\end{aligned} \tag{D.1}$$

Above, ① is because $\mathsf{z}_k^{(i)} = \operatorname{argmin}_{z\in\Delta_{\mathsf{simplex}}}\left\{V_{\mathsf{z}_{k-1}}(z) + \langle(1+\gamma)\alpha_k n\xi_k^{(i)}, z\rangle\right\}$, which is equivalent to saying

$$\begin{aligned} &\forall u \in \Delta_{\mathsf{simplex}}, \quad \langle \nabla V_{\mathsf{z}_{k-1}}(\mathsf{z}_k^{(i)}) + (1+\gamma)\alpha_k n\xi_k^{(i)}, u - \mathsf{z}_k^{(i)}\rangle \geq 0 \\ \iff\ &\forall u \in \Delta_{\mathsf{simplex}}, \quad \langle \nabla w(\mathsf{z}_k^{(i)}) - \nabla w(\mathsf{z}_{k-1}) + (1+\gamma)\alpha_k n\xi_k^{(i)}, u - \mathsf{z}_k^{(i)}\rangle \geq 0. \end{aligned}$$

In particular, we have $\mathbb{1}^T\frac{u^*}{1+\gamma} = \mathbb{1}^T\frac{(1+\varepsilon/2)x^*}{1+\gamma} < 2\mathsf{OPT} \leq 2\mathsf{OPT}'$ and therefore $\frac{u^*}{1+\gamma} \in \Delta_{\mathsf{simplex}}$. Substituting $u = \frac{u^*}{1+\gamma}$ into the above inequality we get ①.

Next, we upper bound the term in the box:

$$\begin{aligned} &(1+\gamma)\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - \mathsf{z}_k^{(i)}\rangle - V_{\mathsf{z}_{k-1}}(\mathsf{z}_k^{(i)}) \\ &\overset{①}{\leq} (1+\gamma)\alpha_k n\xi_{k,i}\cdot(\mathsf{z}_{k-1,i} - \mathsf{z}_{k,i}^{(i)}) - \left(\mathsf{z}_{k,i}^{(i)}\log\frac{\mathsf{z}_{k,i}^{(i)}}{\mathsf{z}_{k-1,i}} + \mathsf{z}_{k-1,i} - \mathsf{z}_{k,i}^{(i)}\right) \\ &\overset{②}{\leq} (1+\gamma)\alpha_k n\xi_{k,i}\cdot(\mathsf{z}_{k-1,i} - \mathsf{z}_{k,i}^{(i)}) - \frac{|\mathsf{z}_{k,i}^{(i)} - \mathsf{z}_{k-1,i}|^2}{2\max\{\mathsf{z}_{k,i}^{(i)}, \mathsf{z}_{k-1,i}\}} \\ &\overset{③}{\leq} (1+\gamma)\alpha_k n\xi_{k,i}\cdot(\mathsf{z}_{k-1,i} - \mathsf{z}_{k,i}^{(i)}) - \frac{|\mathsf{z}_{k,i}^{(i)} - \mathsf{z}_{k-1,i}|^2}{4\mathsf{z}_{k-1,i}} \\ &\overset{④}{\leq} (1+\gamma)^2\mathsf{z}_{k-1,i}\cdot(\alpha_k n\xi_{k,i})^2 \overset{⑤}{\leq} 2\mathsf{z}_{k-1,i}\cdot(\alpha_k n\xi_{k,i})^2 \overset{⑥}{\leq} \mathsf{z}_{k-1,i}\cdot\gamma\alpha_k n|\xi_{k,i}| \\ &\overset{⑦}{\leq} \mathsf{z}_{k-1,i}\cdot\gamma\alpha_k n\xi_{k,i} + 2\mathsf{z}_{k-1,i}\cdot\gamma\alpha_k n\beta = \gamma\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1}\rangle + 2\mathsf{z}_{k-1,i}\cdot\gamma\alpha_k n\beta. \end{aligned} \tag{D.2}$$

Above, ① uses the facts (i) $a\log\frac{a}{b} + b - a \geq 0$ for any $a, b > 0$, (ii) $\mathsf{z}_{k-1,i} - \mathsf{z}_k^{(i)}$ and $\xi_{k,i}$ have the same sign, and (iii) $\xi_{k,i'}^{(i)} = 0$ for every $i' \neq i$; ② uses the inequality that for every $a, b > 0$, we have $a\log\frac{a}{b} + b - a \geq \frac{(a-b)^2}{2\max\{a,b\}}$. ③ uses the fact that $\mathsf{z}_{k,i}^{(i)} \leq 2\mathsf{z}_{k-1,i}$.[14] ④ uses Cauchy-Shwarz: $ab - b^2/4 \leq a^2$. ⑤ uses $(1+\gamma)^2 < 2$. ⑥ uses $|\xi_{k,i}| \leq 1$ and $\gamma = 2\alpha_T n \geq 2\alpha_k n$. ⑦ uses $\xi_{k,i} \geq -\beta$.

Next, we combine (D.1) and (D.2) to conclude that

$$\alpha_k\left\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\right\rangle \leq V_{\mathsf{z}_{k-1}}\left(\frac{u^*}{1+\gamma}\right) - V_{\mathsf{z}_k^{(i)}}\left(\frac{u^*}{1+\gamma}\right) + 2\mathsf{z}_{k-1,i}\cdot\gamma\alpha_k n\beta.$$

Taking expectation on both sides with respect to i, and using the property that $\mathbb{1}^T\mathsf{z}_{k-1} \leq 3\mathsf{OPT}' \leq 6\mathsf{OPT}$, we obtain that

[14] This is because, our parameter choices ensure that $(1+\gamma)\alpha_k n < 1/2\beta$, which further means $-(1+\gamma)\alpha_k n\xi_{k,i}^{(i)} \leq 1/2$. As a result, we must have $\mathsf{z}_{k,i}^{(i)} \leq \mathsf{z}_{k-1,i}\cdot e^{0.5} < 2\mathsf{z}_{k-1,i}$ (see the explicit definition of the mirror step at Proposition 6.4).

$$\mathbf{E}_i\left[\alpha_k\langle n\xi_k^{(i)}, \mathsf{z}_{k-1} - u^*\rangle\right] \le V_{\mathsf{z}_{k-1}}\left(\frac{u^*}{1+\gamma}\right) - \mathbf{E}_i\left[V_{\mathsf{z}_k^{(i)}}\left(\frac{u^*}{1+\gamma}\right)\right] + 12\mathsf{OPT}\cdot\gamma\alpha_k\beta.$$

□

Lemma 6.14 *For every $i \in [n]$, we have*

(a) $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge 0$, *and*
(b) $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge \frac{\mu\beta}{12}\cdot\langle -\widetilde{\eta}_k^{(i)}, u^*\rangle$.

Proof of Lemma 6.14 part (a) Since if $i \notin B_k$ is not a large index we have $\mathsf{y}_k^{(i)} = \mathsf{x}_k$ and the claim is trivial, we focus on $i \in B_k$ in the remaining proof. Recall that $\mathsf{y}_k^{(i)} = \mathsf{x}_k + \delta\mathbf{e}_i$ for some $\delta > 0$ defined in Algorithm 3, so we have

$$f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) = \int_{\tau=0}^{\delta}\langle -\nabla f_\mu(\mathsf{x}_k + \tau\mathbf{e}_i), \mathbf{e}_i\rangle d\tau = \int_{\tau=0}^{\delta}\big(\langle A_{:i}, p(\mathsf{x}_k + \tau\mathbf{e}_i)\rangle - 1\big)d\tau.$$

It is clear that $\langle A_{:i}, p(\mathsf{x}_k + \tau\mathbf{e}_i)\rangle$ decreases as τ increases, and therefore it suffices to prove that $\langle A_{:i}, p(\mathsf{x}_k + \delta\mathbf{e}_i)\rangle \ge 1$.

Suppose that the rows of $A_{:i}$ are sorted (for the simplicity of notation) by the increasing order of $A_{j,i}$. Now, by the definition of the algorithm (recall (6.1)), there exists some $j^* \in [m]$ satisfying that

$$\sum_{j<j^*} A_{j,i}\cdot p_j(\mathsf{x}_k) < 1+\beta \quad \text{and} \quad \sum_{j\le j^*} A_{j,i}\cdot p_j(\mathsf{x}_k) \ge 1+\beta.$$

Next, by our choice of δ which satisfies $\delta = \frac{\mu\beta}{2A_{j^*,i}} \le \frac{\mu\beta}{2A_{j,i}}$ for every $j \le j^*$, we have for every $j \le j^*$:

$$p_j(\mathsf{x}_k + \delta\mathbf{e}_i) = p_j(\mathsf{x}_k)\cdot e^{-\frac{A_{j,i}\delta}{\mu}} \ge p_j(\mathsf{x}_k)\cdot e^{-\beta/2} \ge p_j(\mathsf{x}_k)\cdot(1-\beta/2),$$

and as a result,

$$\begin{aligned}\langle A_{:i}, p(\mathsf{x}_k + \delta\mathbf{e}_i) &\ge \sum_{j\le j^*} A_{j,i}\cdot p_j(\mathsf{x}_k + \delta\mathbf{e}_i) \ge (1-\beta/2)\sum_{j\le j^*} A_{j,i}\cdot p_j(\mathsf{x}_k)\\ &\ge (1-\beta/2)(1+\beta) \ge 1.\end{aligned}$$

□

Proof of Lemma 6.14 part (b) Owing to part (a), for every coordinate i such that $\widetilde{\eta}_{k,i} \ge 0$, we automatically have $f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \ge 0$ so the lemma is obvious. Therefore, let us focus only on coordinates i such that $\widetilde{\eta}_{k,i} < 0$; these are necessarily large indices $i \in B$. Recall from Definition 6.11 that $\widetilde{\eta}_{k,i} = (1+\beta) - (\widetilde{A}^T p(\mathsf{x}_k))_i$, so we have

$$\textstyle\sum_{j=1}^m \widetilde{A}_{j,i}\cdot p_j(\mathsf{x}_k) - (1+\beta) > 0.$$

For the simplicity of description, suppose again that each i-th column is sorted in non-decreasing order, that is, $A_{1,i} \leq \cdots \leq A_{m,i}$. The definition of j^* can be simplified as

$$\sum_{j<j^*} A_{j,i} \cdot p_j(\mathbf{x}_k) < 1+\beta \quad \text{and} \quad \sum_{j\leq j^*} A_{j,i} \cdot p_j(\mathbf{x}_k) \geq 1+\beta.$$

Let $j^\flat \in [m]$ be the row such that

$$\sum_{j<j^\flat} \widetilde{A}_{j,i} \cdot p_j(\mathbf{x}_k) < 1+\beta \quad \text{and} \quad \sum_{j\leq j^\flat} \widetilde{A}_{j,i} \cdot p_j(\mathbf{x}_k) \geq 1+\beta.$$

Note that such a $j^\flat$ must exist because $\sum_{j=1}^m \widetilde{A}_{j,i} \cdot p_j > 1+\beta$. It is clear that $j^\flat \geq j^*$, owing to the definition that $\widetilde{A}_{ji} \leq A_{ji}$ for all $i \in [n]$, $j \in [m]$. Defining $\delta^\flat = \frac{\mu\beta}{2A_{j^\flat,i}} \leq \delta$, the objective decrease is lower bounded as

$$\begin{aligned}
f_\mu(\mathbf{x}_k) - f_\mu(\mathbf{y}_k^{(i)}) &= \int_{\tau=0}^{\delta} \langle -\nabla f_\mu(\mathbf{x}_k+\tau\mathbf{e}_i), \mathbf{e}_i\rangle d\tau = \int_{\tau=0}^{\delta} \big(\langle A_{:i},\, p(\mathbf{x}_k+\tau\mathbf{e}_i)\rangle - 1\big) d\tau \\
&\geq \int_{\tau=0}^{\delta^\flat} \big(\langle A_{:i},\, p(\mathbf{x}_k+\tau\mathbf{e}_i)\rangle - 1\big) d\tau \\
&= \underbrace{\int_{\tau=0}^{\delta^\flat} \left(-1 + \sum_{j\leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k+\tau\mathbf{e}_i)\right) d\tau}_{I} \\
&\quad + \underbrace{\sum_{j>j^\flat} \int_{\tau=0}^{\delta^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k+\tau\mathbf{e}_i) d\tau}_{I'}
\end{aligned}$$

where the inequality is because $\delta^\flat \leq \delta$ and $\langle A_{:i},\, p(\mathbf{x}_k+\tau\mathbf{e}_i)\rangle \geq 1$ for all $\tau \leq \delta$ (see the proof of part (a)).

Part I To lower bound I, we use the monotonicity of $p_j(\cdot)$ and obtain that

$$\begin{aligned}
I &= \int_{\tau=0}^{\delta^\flat} \left(-1 + \sum_{j\leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k+\tau\mathbf{e}_i)\right) d\tau \geq \delta^\flat \\
&\quad \cdot \left(-1 + \sum_{j\leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k+\delta^\flat\mathbf{e}_i)\right).
\end{aligned}$$

However, our choice of $\delta^\flat = \frac{\mu\beta}{2A_{j^\flat,i}} \leq \frac{\mu\beta}{2A_{j,i}}$ for all $j \leq j^\flat$ ensures that

$$\begin{aligned}
\sum_{j\leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k+\delta^\flat\mathbf{e}_i) &\geq \sum_{j\leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k) \cdot e^{\frac{-A_{j,i}\cdot\delta^\flat}{\mu}} \\
&\geq \sum_{j\leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k) \cdot (1-\beta/2).
\end{aligned}$$

Therefore, we obtain that

$$I \geq \delta^\flat \left(-1 + (1-\beta/2) \sum_{j \leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k) \right) \geq \frac{\delta^\flat}{3} \left(-1 + \sum_{j \leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k) \right),$$

where the inequality is because $\left(\frac{2}{3} - \frac{\beta}{2}\right) \sum_{j \leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k) \geq \frac{4-3\beta}{6} \cdot (1+\beta) \geq \frac{2}{3}$ whenever $\beta \leq \frac{1}{3}$ (or equivalently, whenever $\varepsilon \leq 1/9$).

Now, suppose that $\sum_{j \leq j^\flat} \widetilde{A}_{j,i} \cdot p_j(\mathbf{x}_k) - (1+\beta) = b \cdot \widetilde{A}_{j^\flat,i} \cdot p_{j^\flat}(\mathbf{x}_k)$ for some $b \in [0,1]$. Note that we can do so by the very definition of $j^\flat$. Then, we must have

$$\begin{aligned} -1 + \sum_{j \leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k) &\geq -1 + \sum_{j < j^\flat} \widetilde{A}_{j,i} \cdot p_j(\mathbf{x}_k) + A_{j^\flat,i} \cdot p_{j^\flat}(\mathbf{x}_k) \\ &= -1 + (1+\beta) - (1-b)\widetilde{A}_{j^\flat,i} \cdot p_{j^\flat}(\mathbf{x}_k) + A_{j^\flat,i} \cdot p_{j^\flat}(\mathbf{x}_k) \\ &\geq \beta + b \cdot A_{j^\flat,i} \cdot p_{j^\flat}(\mathbf{x}_k). \end{aligned}$$

Therefore, we conclude that

$$\begin{aligned} I &\geq \frac{\delta^\flat}{3} \left(-1 + \sum_{j \leq j^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k) \right) > \frac{\delta^\flat}{3} \cdot b \cdot A_{j^\flat,i} \cdot p_{j^\flat}(\mathbf{x}_k) \\ &= \frac{\mu\beta}{6\widetilde{A}_{j^\flat,i}} \cdot b \cdot \widetilde{A}_{j^\flat,i} \cdot p_{j^\flat}(\mathbf{x}_k) \\ &= \frac{\mu\beta}{6\widetilde{A}_{j^\flat,i}} \cdot \left(-(1+\beta) + \sum_{j \leq j^\flat} \widetilde{A}_{j,i} \cdot p_j(\mathbf{x}_k) \right) \\ &\geq \frac{\mu\beta}{12} \cdot u_i^* \cdot \left(-(1+\beta) + \sum_{j \leq j^\flat} \widetilde{A}_{j,i} \cdot p_j(\mathbf{x}_k) \right). \end{aligned}$$

Above, the last inequality is because $u_i^* \cdot \widetilde{A}_{j^\flat,i} \leq \langle \widetilde{A}_{j^\flat,:}, u^* \rangle \leq 2$ by our definition of $\widetilde{A}$.

Part I' To lower bound I', consider every $j > j^\flat$ and the integral

$$\int_{\tau=0}^{\delta^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k + \tau \mathbf{e}_i) d\tau.$$

Note that whenever $\tau \leq \frac{\mu\beta}{2A_{j,i}} \leq \frac{\mu\beta}{2A_{j^\flat,i}} = \delta^\flat$, we have that $p_j(\mathbf{x}_k + \tau\mathbf{e}_i) \geq p_j(\mathbf{x}_k) \cdot e^{-\beta/2} \geq \frac{1}{2} p_j(\mathbf{x}_k)$. Therefore,

$$\int_{\tau=0}^{\delta^\flat} A_{j,i} \cdot p_j(\mathbf{x}_k + \tau\mathbf{e}_i) d\tau \geq \int_{\tau=0}^{\frac{\mu\beta}{2A_{j,i}}} A_{j,i} \cdot p_j(\mathbf{x}_k + \tau\mathbf{e}_i) d\tau \geq \frac{\mu\beta}{2A_{j,i}} \cdot A_{j,i} \cdot \frac{1}{2} p_j(\mathbf{x}_k).$$

This implies a lower bound on I':

$$I' \geq \sum_{j>j^\flat} \frac{\mu\beta}{4A_{j,i}} \cdot A_{j,i} \cdot p_j(\mathsf{x}_k) \geq \frac{\mu\beta}{8} \cdot \sum_{j>j^\flat} u_i^* \cdot \widetilde{A}_{j,i} \cdot p_j(\mathsf{x}_k),$$

where again in the last inequality we have used $u_i^* \cdot \widetilde{A}_{j^\flat,i} \leq \langle \widetilde{A}_{j^\flat:}, u^* \rangle \leq 2$ by our definition of $\widetilde{A}$.

Together Combining the lower bounds on I and I', we obtain

$$\begin{aligned} f_\mu(\mathsf{x}_k) - f_\mu(\mathsf{y}_k^{(i)}) \geq I + I' &\geq \frac{\mu\beta}{12} \cdot u_i^* \cdot \left(-(1+\beta) + \sum_{j=1}^{m} \widetilde{A}_{j,i} \cdot p_j(\mathsf{x}_k) \right) \\ &= \frac{\mu\beta}{12} \cdot \langle -\widetilde{\eta}_k^{(i)}, u^* \rangle. \end{aligned}$$

□

E.5 Proof of Lemma 3.3: Efficient Implementation of *PacLPSolver*

In this section, we illustrate how to implement each iteration of *PacLPSolver* to run in an expected $O(N/n)$ time. We maintain the following quantities

$$\mathsf{z}_k \in \mathbb{R}^n_{\geq 0}, \quad \mathsf{az}_k \in \mathbb{R}^m_{\geq 0}, \quad \mathsf{y}'_k \in \mathbb{R}^n, \quad \mathsf{ay}'_k \in \mathbb{R}^m, \quad B_{k,1}, B_{k,2} \in \mathbb{R}_+$$

throughout the algorithm, so as to ensure the following invariants are always satisfied

$$A\mathsf{z}_k = \mathsf{az}_k, \tag{E.1}$$

$$\mathsf{y}_k = B_{k,1} \cdot \mathsf{z}_k + B_{k,2} \cdot \mathsf{y}'_k, \quad A\mathsf{y}'_k = \mathsf{ay}'_k. \tag{E.2}$$

It is clear that when $k = 0$, letting $\mathsf{az}_k = A\mathsf{z}_0$, $\mathsf{y}'_k = \mathsf{y}_0$, $\mathsf{ay}'_k = A\mathsf{y}_0$, $B_{k,1} = 0$, and $B_{k,2} = 1$, we can ensure that all the invariants are satisfied initially. We denote $\|A_{:i}\|_0$ the number of nonzeros elements in vector $A_{:i}$. In each iteration $k = 1, 2, \ldots, T$:

- The step $\mathsf{x}_k = \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$ does not need to be implemented.
- The value $\nabla_i f(\mathsf{x}_k)$ requires the knowledge of $p_j(\mathsf{x}_k) = e^{\frac{1}{\mu}((A\mathsf{x}_k)_j - 1)}$ for each j such that $A_{ij} \neq 0$. Accordingly, for each j, we need to know the value

$$\begin{aligned} (A\mathsf{x}_k)_j &= \tau(A\mathsf{z}_{k-1})_j + (1-\tau)(A\mathsf{y}_{k-1})_j \\ &= \big(\tau + (1-\tau)B_{k-1,1}\big)\mathsf{az}_{k-1,j} + (1-\tau)B_{k-1,2}\mathsf{ay}'_{k-1,j}. \end{aligned}$$

  This can be computed in $O(1)$ time for each j, and $O(\|A_{:i}\|_0)$ time in total.
- Recall the step $\mathsf{z}_k \leftarrow \operatorname{argmin}_{z \in \Delta_{\mathsf{box}}} \left\{ \frac{1}{2}\|z - \mathsf{z}_{k-1}\|_A^2 + \langle n\alpha_k \xi_k^{(i)}, z \rangle \right\}$ can be written as $\mathsf{z}_k = \mathsf{z}_{k-1} + \delta\mathbf{e}_i$ for some $\delta \in \mathbb{R}$ that can be computed in $O(1)$ time (see Proposition 3.2). Observe also $\mathsf{z}_k = \mathsf{z}_{k-1} + \delta\mathbf{e}_i$ yields $\mathsf{y}_k = \tau\mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1} +$

$\frac{\delta \mathbf{e}_i}{n\alpha_k L}$ due to Line 6 and Line 10 of Algorithm 1. Therefore, we perform two explicit updates on z_k and az_k as

$$\mathsf{z}_k \leftarrow \mathsf{z}_{k-1} + \delta \mathbf{e}_i, \quad \mathsf{az}_k \leftarrow \mathsf{az}_{k-1} + \delta A_{:i}$$

and two implicit updates on y_k as

$$B_{k,1} = \tau + (1-\tau)B_{k-1,1},\ B_{k,2} = (1-\tau)B_{k-1,2},$$
$$\mathsf{y}'_k \leftarrow \mathsf{y}'_{k-1} + \delta \mathbf{e}_i \cdot \left(-\tfrac{B_{k,1}}{B_{k,2}} + \tfrac{1}{n\alpha_k L}\tfrac{1}{B_{k,2}}\right), \mathsf{ay}'_k \leftarrow \mathsf{ay}'_{k-1} + \delta A_{:i} \cdot \left(-\tfrac{B_{k,1}}{B_{k,2}} + \tfrac{1}{n\alpha_k L}\tfrac{1}{B_{k,2}}\right)$$

It is not hard to verify that after these updates, $A\mathsf{y}'_k = \mathsf{ay}'_k$ and we have

$$\begin{aligned}
B_{k,1} \cdot \mathsf{z}_k + B_{k,2} \cdot \mathsf{y}'_k &= B_{k,1} \cdot \left(\mathsf{z}_{k-1} + \delta \mathbf{e}_i\right) \\
&\quad + B_{k,2} \cdot \left(\mathsf{y}'_{k-1} + \delta \mathbf{e}_i \cdot \left(-\frac{B_{k,1}}{B_{k,2}} + \frac{1}{n\alpha_k L}\frac{1}{B_{k,2}}\right)\right) \\
&= B_{k,1} \cdot \mathsf{z}_{k-1} + B_{k,2} \cdot \left(\mathsf{y}'_{k-1} + \delta \mathbf{e}_i \cdot \left(\frac{1}{n\alpha_k L}\frac{1}{B_{k,2}}\right)\right) \\
&= B_{k,1} \cdot \mathsf{z}_{k-1} + B_{k,2} \cdot \mathsf{y}'_{k-1} + \frac{\delta \mathbf{e}_i}{n\alpha_k L} \\
&= \left(\tau + (1-\tau)B_{k-1,1}\right) \cdot \mathsf{z}_{k-1} + \left((1-\tau)B_{k-1,2}\right) \cdot \mathsf{y}'_{k-1} + \frac{\delta \mathbf{e}_i}{n\alpha_k L} \\
&= \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1} + \frac{\delta \mathbf{e}_i}{n\alpha_k L} = \mathsf{y}_k,
\end{aligned}$$

so the invariant $\mathsf{y}_k = B_{k,1} \cdot \mathsf{z}_k + B_{k,2} \cdot \mathsf{y}'_k$ also holds. In sum, after performing updates on $A\mathsf{z}_k$ and ay'_k in time $O(\|A_{:i}\|_0)$, we can ensure that the invariants in (E.1) and (E.2) are satisfied at iteration k.

In sum, we only need $O(\|A_{:i}\|_0)$ time to perform the updates in *PacLPSolver* for an iteration k if the coordinate i is selected. Therefore, each iteration of *PacLPSolver* can be implemented to run in an expected $O(\mathbf{E}_i[\|A_{:i}\|_0]) = O(N/n)$ time.

F.6 Proof of Lemma 6.5: Efficient Implementation of *CovLPSolver*

In this section we illustrate how to implement each iteration of *CovLPSolver* to run in an expected $O(N/n)$ time. We maintain the following quantities

$$\mathsf{z}'_k \in \mathbb{R}^n_+, \quad \mathsf{sz}_k \in \mathbb{R}_+, \quad \mathsf{sumz}_k \in \mathbb{R}_+, \quad \mathsf{az}'_k \in \mathbb{R}^m_{\geq 0}, \quad \mathsf{y}'_k \in \mathbb{R}^n,$$
$$\mathsf{ay}'_k \in \mathbb{R}^m, \quad B_{k,1}, B_{k,2} \in \mathbb{R}_+$$

throughout the algorithm, so as to maintain the following invariants

$$\mathsf{z}_k = \mathsf{z}'_k/\mathsf{sz}_k, \qquad \mathsf{sumz}_k = \mathbb{1}^T \mathsf{z}'_k, \qquad A\mathsf{z}_k = \mathsf{az}'_k/\mathsf{sz}_k, \tag{F.1}$$
$$\mathsf{y}_k = B_{k,1} \cdot \mathsf{z}'_k + B_{k,2} \cdot \mathsf{y}'_k, \qquad A\mathsf{y}_k = \mathsf{ay}'_k. \tag{F.2}$$

It is clear that when $k = 0$, letting $\mathsf{z}'_k = \mathsf{z}_0$, $\mathsf{sz}_k = 1$, $\mathsf{sumz}_k = \mathbb{1}^T \mathsf{z}_0$, $\mathsf{az}'_k = A\mathsf{z}_0$, $\mathsf{y}'_k = \mathsf{y}_0$, $\mathsf{ay}'_k = A\mathsf{y}_0$, $B_{k,1} = 0$, and $B_{k,2} = 1$, we can ensure that all the invariants are satisfied initially.

We denote by $\|A_{:i}\|_0$ the number of nonzero elements in vector $A_{:i}$. In each iteration $k = 1, 2, \ldots, T$:

- The step $\mathsf{x}_k = \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1}$ does not need to be implemented.
- The value $p_j(\mathsf{x}_k) = e^{\frac{1}{\mu}(1-(A\mathsf{x}_k)_j)}$ for each j only requires the knowledge of

$$\begin{aligned}(A\mathsf{x}_k)_j &= \tau(A\mathsf{z}_{k-1})_j + (1-\tau)(A\mathsf{y}_{k-1})_j \\ &= \big(\tau + (1-\tau)B_{k-1,1}\big)\frac{\mathsf{az}'_{k-1,j}}{\mathsf{sz}_{k-1}} + (1-\tau)B_{k-1,2}\mathsf{ay}'_{k-1,j}.\end{aligned}$$

  This can be computed in $O(1)$ time.
- The value $\nabla_i f(\mathsf{x}_k)$ requires the knowledge of $p_j(\mathsf{x}_k)$ for each $j \in [m]$ such that $A_{ij} \neq 0$. Since we have $\|A_{:i}\|_0$ such j's, we can compute $\nabla_i f(\mathsf{x}_k)$ in $O(\|A_{:i}\|_0)$ time.
- Letting $\delta = (1+\gamma)n\alpha_k \xi^{(i)}_{k,i}$, recall that the mirror step $\mathsf{z}_k \leftarrow \operatorname{argmin}_{z \in \Delta_{\mathsf{simplex}}} \left\{V_{\mathsf{z}_{k-1}}(z) + \langle \delta \mathbf{e}_i, z\rangle\right\}$ has a very simple form (see Proposition 6.4): first multiply the i-th coordinate of z_{k-1} by $e^{-\delta}$ and then, if the sum of all coordinates have exceeded $2\mathsf{OPT}'$, scale everything down so as to sum up to $2\mathsf{OPT}'$. This can be implemented as follows: setting $\delta_1 = \mathsf{z}'_{k-1,i}(e^{-\delta} - 1)$,

$$\begin{aligned}&\mathsf{z}'_k \leftarrow \mathsf{z}'_{k-1} + \delta_1 \mathbf{e}_i, \mathsf{az}'_k \leftarrow \mathsf{az}'_{k-1} + \delta_1 A_{:i}, \\ &\mathsf{sumz}_k \leftarrow \mathsf{sumz}_{k-1} + \delta_1, \mathsf{sz}_k \leftarrow \mathsf{sz}_k \cdot \max\left\{1, \tfrac{\mathsf{sumz}_k}{\mathsf{sz}_{k-1} \cdot 2\mathsf{OPT}'}\right\}.\end{aligned}$$

  These updates can be implemented to run in $O(\|A_{:i}\|_0)$ time, and they together ensure that the invariants in (F.1) are satisfied at iteration k.
- Recall that the gradient step is of the form $\mathsf{y}_k \leftarrow \mathsf{x}_k + \delta_2 \cdot \mathbf{e}_i$ for some value $\delta_2 \geq 0$. This value δ_2 can be computed in $O(\|A_{:i}\|_0)$ time, since each $p_j(\mathsf{x}_k)$ can be computed in $O(1)$ time, and we can sort the rows of each column of A by preprocessing.

  Since $\mathsf{y}_k = \mathsf{x}_k + \delta_2 \cdot \mathbf{e}_i = \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1} + \delta_2 \mathbf{e}_i$, we can implement this update by letting

$$\begin{aligned}&B_{k,1} = \tfrac{\tau}{\mathsf{sz}_{k-1}} + (1-\tau)B_{k-1,1}, B_{k,2} = (1-\tau)B_{k-1,2} \\ &\mathsf{y}'_k \leftarrow \mathsf{y}'_{k-1} + \mathbf{e}_i \cdot \left(-\tfrac{B_{k,1}\delta_1}{B_{k,2}} + \tfrac{\delta_2}{B_{k,2}}\right), \mathsf{ay}'_k \leftarrow \mathsf{ay}'_{k-1} + A_{:i} \cdot \left(-\tfrac{B_{k,1}\delta_1}{B_{k,2}} + \tfrac{\delta_2}{B_{k,2}}\right)\end{aligned}$$

  It is not hard to verify that after these updates, $\mathsf{ay}'_k = A\mathsf{y}'_k$ and we have

$$\begin{aligned}B_{k,1} \cdot \mathsf{z}'_k + B_{k,2} \cdot \mathsf{y}'_k &= B_{k,1} \cdot \big(\mathsf{z}'_{k-1} + \delta_1 \mathbf{e}_i\big) \\ &\quad + B_{k,2} \cdot \left(\mathsf{y}'_{k-1} + \mathbf{e}_i \cdot \left(-\frac{B_{k,1}\delta_1}{B_{k,2}} + \frac{\delta_2}{B_{k,2}}\right)\right) \\ &= B_{k,1} \cdot \mathsf{z}'_{k-1} + B_{k,2} \cdot \big(\mathsf{y}'_{k-1} + \delta_2 \mathbf{e}_i / B_{k,2}\big)\end{aligned}$$

$$= B_{k,1} \cdot \mathsf{z}'_{k-1} + B_{k,2} \cdot \mathsf{y}'_{k-1} + \delta_2 \mathbf{e}_i$$
$$= \Big(\frac{\tau}{\mathsf{s}\mathsf{z}_{k-1}} + (1-\tau)B_{k-1,1}\Big) \cdot \mathsf{z}'_{k-1} + \big((1-\tau)B_{k-1,2}\big) \cdot \mathsf{y}'_{k-1} + \delta_2 \mathbf{e}_i$$
$$= \tau \mathsf{z}_{k-1} + (1-\tau)\mathsf{y}_{k-1} + \delta_2 \mathbf{e}_i = \mathsf{y}_k,$$

so that the invariant $\mathsf{y}_k = B_{k,1} \cdot \mathsf{z}'_k + B_{k,2} \cdot \mathsf{y}'_k$ is also satisfied. In sum, after running time $O(\|A_{:i}\|_0)$, we can ensure that the invariants in (F.2) are satisfied at iteration k.

In sum, we only need $O(\|A_{:i}\|_0)$ time to perform the updates in *CovLPSolver* for an iteration k if the coordinate i is selected. Therefore, each iteration of *CovLPSolver* can be implemented to run in an expected $O(\mathbf{E}_i[\|A_{:i}\|_0]) = O(N/n)$ time.

References

1. Allen-Zhu, Z., Lee, Y.T., Orecchia, L.: Using optimization to obtain a width-independent, parallel, simpler, and faster positive SDP solver. In: SODA (2016)
2. Allen-Zhu, Z., Li, Y., Oliveira, R., Wigderson, A: Much faster algorithms for matrix scaling. In: FOCS, 2017. arXiv:1704.02315
3. Allen-Zhu, Z., Liao, Z., Orecchia, L.: Spectral sparsification and regret minimization beyond multiplicative updates. In: STOC (2015)
4. Allen-Zhu, Z., Orecchia, L.: Using optimization to break the epsilon barrier: a faster and simpler width-independent algorithm for solving positive linear programs in parallel. In: SODA (2015)
5. Allen-Zhu, Z., Orecchia, L.: Linear coupling: an ultimate unification of gradient and mirror descent. In: ITCS (2017)
6. Allen-Zhu, Z., Qu, Z., Richtárik, P., Yuan, Y.: Even faster accelerated coordinate descent using non-uniform sampling. In: ICML (2016)
7. Arora, S., Hazan, E., Kale, S.: The multiplicative weights update method: a meta-algorithm and applications. Theory Comput. **8**, 121–164 (2012)
8. Awerbuch, B., Khandekar, R.: Stateless distributed gradient descent for positive linear programs. In: STOC (2008)
9. Awerbuch, B., Khandekar, R., Rao, S.: Distributed algorithms for multicommodity flow problems via approximate steepest descent framework. ACM Trans. Algorithms **9**(1), 1–14 (2012)
10. Bartal, Y., Byers, J.W., Raz, D.: Global optimization using local information with applications to flow control. In: Proceedings 38th Annual Symposium on Foundations of Computer Science, pp. 303–312. IEEE Computer Society (1997)
11. Bartal, Y., Byers, J.W., Raz, D.: Fast, distributed approximation algorithms for positive linear programming with applications to flow control. SIAM J. Comput. **33**(6), 1261–1279 (2004)
12. Ben-Tal, A., Arkadi, N.: Lectures on modern convex optimization. Soc. Ind. Appl. Math. 315–341 (2013)
13. Bienstock, D., Iyengar, G.: Faster approximation algorithms for packing and covering problems. Technical report, Columbia University, September 2004. Preliminary version published in STOC '04
14. Byers, J., Nasser, G.: Utility-based decision-making in wireless sensor networks. In: 2000 first annual workshop on mobile and ad hoc networking and computing, 2000. MobiHOC, pp. 143–144. IEEE (2000)
15. Chudak, F.A., Eleutério, V. : Improved approximation schemes for linear programming relaxations of combinatorial optimization problems. In: Proceedings of the 11th International IPCO Conference on Integer Programming and Combinatorial Optimization, pp. 81–96 (2005)
16. Duan, R., Pettie, S.: Linear-time approximation for maximum weight matching. J. ACM **61**(1), 1–23 (2014)
17. Fercoq, O., Richtárik, P.: Accelerated, parallel and proximal coordinate descent. SIAM J. Optim. **25**(4), 1997–2023 (2015)
18. Fleischer, L.K.: Approximating fractional multicommodity flow independent of the number of commodities. SIAM J. Discrete Math. **13**(4), 505–520 (2000)

19. Garg, N., Könemann, J.: Faster and simpler algorithms for multicommodity flow and other fractional packing problems. SIAM J. Comput. **37**(2), 630–652 (2007)
20. Grigoriadis, M.D., Khachiyan, L.G.: Fast approximation schemes for convex programs with many blocks and coupling constraints. SIAM J. Optim. **4**(1), 86–107 (1994)
21. Jain, R., Ji, Z., Upadhyay, S., Watrous, J.: QIP = PSPACE. J. ACM JACM **58**(6), 30 (2011)
22. Klein, P., Young, N.E.: On the number of iterations for Dantzig–Wolfe optimization and packing-covering approximation algorithms. SIAM J. Comput. **44**(4), 1154–1172 (2015)
23. Koufogiannakis, C., Young, N.E.: A nearly linear-time PTAS for explicit fractional packing and covering linear programs. Algorithmica **70**, 494–506 (2013). **(Previously appeared in FOCS '07)**
24. Luby, M., Nisan, N.: A parallel approximation algorithm for positive linear programming. In: STOC, pp. 448–457. ACM Press, New York (1993)
25. Madry, A.: Faster approximation schemes for fractional multicommodity flow problems via dynamic graph algorithms. In: STOC. ACM Press, New York (2010)
26. Nemirovski, A.: Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems. SIAM J. Optim. **15**(1), 229–251 (2004)
27. Nesterov, Y.: Rounding of convex sets and efficient gradient methods for linear programming problems. Optim. Methods Softw. **23**(1), 109–128 (2008)
28. Nesterov, Y.: A method of solving a convex programming problem with convergence rate $O(1/k^2)$. Sov. Math. Dokl. **269**, 543–547 (1983)
29. Nesterov, Y.: Smooth minimization of non-smooth functions. Math. Program. **103**(1), 127–152 (2005)
30. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM J Optim. **22**(2), 341–362 (2012)
31. Plotkin, S.A., Shmoys, D.B., Tardos, É.: Fast approximation algorithms for fractional packing and covering problems. Math. Oper. Res. **20**(2), 257–301 (1995). **(conference version published in FOCS 1991)**
32. Trevisan, L.: Parallel approximation algorithms by positive linear programming. Algorithmica **21**(1), 72–88 (1998)
33. Wang, D., Mahoney, M., Mohan, N., Rao, S.: Faster parallel solver for positive linear programs via dynamically-bucketed selective coordinate descent. ArXiv e-prints arXiv:1511.06468 (2015)
34. Wang, D., Rao, S., Mahoney, M.W.: Unified acceleration method for packing and covering problems via diameter reduction. In: ICALP (2016)
35. Young, N.E.: Sequential and parallel algorithms for mixed packing and covering. In: 42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS'01), pp. 538–546. IEEE Computer Society (2001)
36. Young, N.E.: Nearly linear-time approximation schemes for mixed packing/covering and facility-location linear programs. ArXiv e-prints arXiv:1407.3015 (2014)
37. Zurel, E., Nisan, N.: An efficient approximate allocation algorithm for combinatorial auctions. In: Proceedings of the 3rd ACM Conference on Electronic Commerce, pp. 125–136. ACM (2001)