Primal-Dual Stochastic Mirror Descent for MDPs

Daniil Tiapkin
HSE University, Russia

Alexander Gasnikov
Moscow Institute of Physics and Technology, Russia
HSE Univeristy, Russia

Abstract

We consider the problem of learning the optimal policy for infinite-horizon Markov decision processes (MDPs). For this purpose, some variant of Stochastic Mirror Descent is proposed for convex programming problems with Lipschitz-continuous functionals. An important detail is the ability to use inexact values of functional constraints and compute the value of dual variables. We analyze this algorithm in a general case and obtain an estimate of the convergence rate that does not accumulate errors during the operation of the method. Using this algorithm, we get the first parallel algorithm for mixing average-reward MDPs with a generative model without reduction to discounted MDP. One of the main features of the presented method is low communication costs in a distributed centralized setting, even with very large networks.

1 INTRODUCTION

We consider the following nonsmooth convex optimization problem over a simple closed convex set $Q \subseteq E$, where E is a finite-dimensional normed space, with additional functional constraints

$$\begin{aligned}&\min_{x\in Q} f(x),\\ &\text{s.t. } g^{(l)}(x) \le 0,\ \forall l \in [m].\end{aligned}$$

In the context of modern large-scale optimization, the number of constraints m could be huge, so we are interested in first-order algorithms in which a number of iterations does not depend on m. In book by Nemirovski and Yudin (1983) it was noticed that the usual (sub)gradient method could handle functional constraints without additional price in terms of a total number of iterations. The proposed scheme in the simple form could be written as follows

$$x^{k+1} = \begin{cases} x^k - \eta \nabla f(x^k), & \max_{l\in[m]} g^{(l)}(x^k) \le \varepsilon;\\ x^k - \eta \nabla g^{(l(k))}(x^k), & \max_{l\in[m]} g^{(l)}(x^k) > \varepsilon,\end{cases}$$

where $l(k) = \operatorname{argmax}_{l\in[m]} g^{(l)}(x^k)$ and ε is a desired accuracy of constraint satisfaction. We emphasize that this scheme could be paralleled very efficiently: different threads or nodes of the network could handle the computation of different constraints. The ability to parallel computation is crucial for any large-scale application.

Next there were two main directions in the development of this scheme:

- use of stochastic (sub)gradients;
- computation of dual variables for the Lagrange dual problem (primal-duality).

The first direction is essential for large-scale applications because the computation of exact gradients could be impossible or computationally heavy. The value of the second type of development highly depends on the particular application but always gives a possibility to use stopping criteria based on the duality gap.

The development of the stochastic case was initiated in the paper (Nemirovski et al., 2009) for Mirror Descent without functional constraint and developed for high-probability deviations bounds in the paper (Lan et al., 2012). The work (Bayandina et al., 2018) makes possible application of these results to the scheme with functional constraints.

The first step to the primal-duality of subgradient method was done in the paper (Nesterov, 2009) but in a different direction: the author solves the dual problem using subgradient method and reconstructs the primal variables. In the work (Nesterov and Shpirko, 2014) the authors propose the scheme that solves the

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).

primal conic problem using subgradient method with functional constraints and afterward computes dual variables without any computational price. This approach was generalized to arbitrary deterministic convex optimization problems in the paper (Bayandina et al., 2018).

However, a natural question appears: *Is it possible to combine these two properties and propose a stochastic subgradient algorithm that computes the dual variables without additional computational price?* In this paper, we give a positive answer. Additionally, we propose a bright application that requires combining both traits with additional inexactness in constraint computation.

Markov Decision Process. We apply the proposed primal-dual stochastic Mirror Descent to the problem of *mixing average-reward Markov Decision Process.*

Markov Decision Process (MDP) is a mathematical model for the reinforcement learning (RL) problem, the rapidly developing branch of modern machine learning (Sutton and Barto, 2018; Szepesvari, 2010). We consider the infinite horizon average-reward setting of this problem. For complete notations we refer to Section 3.

The solution to the average-reward MDP (AMDP) with S states and A_{tot} state-action pairs could be described through the linear program that obtained from Bellman equations (Bertsekas, 2005)

$$\begin{aligned}&\min_{\bar v, h} \ \bar v\\ &\text{s.t. } \ \bar v\mathbf{1} + (\hat{\mathbf{I}} - \mathbf{P})h - \mathbf{r} \ge 0,\end{aligned}$$

where $\hat{\mathbf{I}}_{(i,a_i),j} = \mathbf{I}_{i,j}, \mathbf{P} \in \mathbb{R}^{A_{\text{tot}}\times S}$ is a transition probability matrix, $\mathbf{r} \in \mathbb{R}^{A_{\text{tot}}}$ is a vector of rewards for state-action pairs, $\bar v$ is an average reward value, and h is a bias vector. For another setting of discounted MDP we refer to the line of the previous work (Azar et al., 2012; Sidford et al., 2018a,b; Agarwal et al., 2020; Li et al., 2020) and references within.

Let us list important properties of the problem (6): 1) it has a huge number of constraints A_{tot}; 2) it is impossible to compute neither constraints nor its gradient since the transition probability matrix $\mathbf{P}$ is usually unknown; 3) a policy that corresponds to the optimal average reward $\bar v$ can be computed from the solution to the dual problem of (6).

To handle the second problem we consider solving AMDP with *generative model* or *sampler* (Azar et al., 2012; Jin and Sidford, 2020; Agarwal et al., 2020): a stochastic oracle generates a transition state from a given state-action pair according to probabilities $\mathcal{P}$. This assumption makes applying of the proposed stochastic primal-dual Mirror Descent possible to the problem (6) after a suitable approximation of constraint functions. We underline that it is required to use a combination of *all* properties of our algorithm to solve this problem. Additionally, we notice that the work (Lan and Zhou, 2020) can handle stochastic constraints without additional approximation of constraint functions but at the price of much worse complexity $\sim \varepsilon^{-4}$.

However, it is impossible to obtain rates for AMDP solving without additional assumptions. We consider the *mixing* assumption that the Markov chain that corresponds to the choice of any policy converges to the stationary distribution sufficiently fast. In papers (Wang, 2017; Jin and Sidford, 2020) authors showed that under mixing assumption, the search space for the bias vector h could be bounded using mixing time t_{mix}. A similar assumption was studied in the paper (Kearns and Singh, 2002) and an alternative one of communicating MDP with a finite diameter in works (Bartlett and Tewari, 2009; Jaksch et al., 2010; Agrawal and Jia, 2017).

To the authors' best knowledge, there are only three works that consider the infinite-horizon mixing AMDP with a generative model – (Wang, 2017; Jin and Sidford, 2020, 2021). In the first two papers the same general convex optimization algorithm was used – Stochastic Mirror Descent for saddle-point problems, and both of the presented algorithms are not designed for parallel computations. In the last paper authors perform reduction to the discounted problem, and the price of this reduction is sample complexity of order $O(\varepsilon^{-3})$. In this paper, we present the first *parallel* algorithm for this problem without reductions to discounted MDP with very low communication costs. The parallelism gives a possibility to handle very large setups of MDPs that cannot be stored in the memory of one machine. In the case of simultaneous working of A_{tot} workers, our algorithm works in $\tilde O(t_{\text{mix}}^2|\mathcal{S}|\varepsilon^{-2})$ real time and outperforms approach of Jin and Sidford (2020) which works in $\tilde O(t_{\text{mix}}^2 A_{\text{tot}}\varepsilon^{-2})$.

Our contribution.

- The first (sub)gradient-based algorithm for optimization with functional constraints that 1) allows the use of stochastic gradients, 2) computes the value of Lagrange dual variables, 3) allows inexact computation of constraints at the same time.
- The first parallel algorithm for solving mixing average-reward MDP without reduction to the discounted problem. Additionally, this algorithm has very low communication costs and thus could

work on very large centralized networks effectively.

Paper organization. Section 2 is devoted to the proposed primal-dual stochastic variant of the Mirror Descent algorithm. All proofs are presented in the supplementary material. Section 3 describes how to apply the results of Section 2 to the mixing AMDP problem. Finally, Section 4 contains a numerical comparison to the approach described in the paper of Jin and Sidford (2020).

Notation. For a matrix $A \in \mathbb{R}^{n\times m}$ we define its i-th row as $A_{(i)}$. We denote by $\mathbf{1} = (1,\dots,1)^\top$ the vector filled with ones. By e_i we define a standard basis vector. Also we define $\Delta^n = \{x \in \mathbb{R}^n \mid \forall i : x_i \geq 0, \sum_{i=1}^n x_i = 1\}$ and $\mathbb{B}^n_c = [-c,c]^n$. $\mathbf{I}$ is an identity matrix of size deducible from the context. Inner product $\langle\cdot,\cdot\rangle\colon E^* \times E \to \mathbb{R}$ is defined on pairs of vectors from dual and primal spaces. In the case of Euclidean spaces, it coincides with the standard inner product. For a normed space $(E, \|\cdot\|)$ we define a dual norm on a space E^* as follows: $\|v\|_* = \sup_{x:\|x\|=1}\langle v, x\rangle$. Also we define $[m] = \{1,\dots,m\}$. By $\mathbb{I}\{A\}$ we define an indicator of a set A. For $i \in [m]$ define $e_i \in \mathbb{R}^m : e_i[j] = \mathbb{I}\{i = j\}$.

2 PRIMAL-DUAL STOCHASTIC MIRROR DESCENT

In this section we develop techniques of Bayandina et al. (2018). Firstly, we introduce a basic notation that will be used further. Then we propose a new algorithm for the constrained convex stochastic optimization problem in the model of inexact computation of constraint functions. Finally, we prove convergence of the algorithm in terms of the duality gap between primal and (Lagrange) dual problems. The last part is crucial for an application on average-reward MDPs.

2.1 Notation

We consider the constrained convex optimization problem over a convex compact set $Q \subseteq E$, where E is a finite-dimensional normed space

$$\begin{aligned} &\min_{x\in Q} f(x), \\ &\text{s.t. } g^{(l)}(x) \leq 0,\ \forall l \in [m], \end{aligned} \tag{1}$$

and $f\colon Q \to \mathbb{R}, g^{(l)}\colon Q \to \mathbb{R}$ are convex functions. We assume that subgradients of these functions exist for each $x \in Q$ for simplicity. We call $\nabla f(x), \nabla g^{(l)}(x)$ any subgradients of corresponding functions. However, in our algorithm we have an access only to stochastic subgradient oracles $\nabla f(x,\xi), \nabla g^{(l)}(x,\xi_{(l)})$.

Now we are going to introduce definitions that will be useful in the algorithm description.

Definition 1 *Function $h_\delta\colon Q \to \mathbb{R}$ is called a δ-approximation of $h\colon Q \to \mathbb{R}$ if $|h_\delta(x) - h(x)| \leq \delta$ for all $x \in Q$.*

For our problem, we suggest that we have an oracle not for computation of constraints $g^{(l)}$ but their δ-approximations $g^{(l)}_\delta$. It is the important difference with the setup of Bayandina et al. (2018): in our assumption we cannot consider only one constraint of form $g(x) = \max_{l\in[m]} g^{(l)}(x)$ because it does not seem possible to compute a subgradient of g given subgradients of $g^{(l)}$ when index on which the maximal value attains is unknown.

Now we are ready to list all our assumptions on problem setup (1) for our algorithm:

(A1) f and all $g^{(l)}$ are Lipschitz continuous with constant M for objective function and for all constraints;

(A2) Stochastic subgradients are unbiased: $\mathbb{E}\nabla f(x,\xi) = \nabla f(x)$, $\mathbb{E}\nabla g^{(l)}(x,\xi_{(l)}) = \nabla g^{(l)}(x)$;

(A3) Stochastic subgradiens are bounded: $\|\nabla f(x,\xi)\|_* \leq M, \|\nabla g^{(l)}(x,\xi_{(l)})\|_* \leq M$ a.s.;

Since our algorithm is Mirror-Descent based, the next step is to define the proximal step and basic properties of Mirror Descent.

Firstly, we define a *prox-function* $d\colon Q \to \mathbb{R}$ as a continuous 1-strongly convex function d with respect to the norm $\|\cdot\|$ on E that admits continous selection of subgradients $\nabla d(x)$ where they exist. *Bregman divergence* that corresponds to a prox-function d is a function $V(x,y) = d(y) - d(x) - \langle\nabla d(x), y - x\rangle$.

Given vectors $x \in E$ and $v \in E^*$, the mirror step is defined as

$$x^+ = \mathrm{Mirr}(x, v) = \operatorname*{argmin}_{y\in X}\left\{\langle v, y\rangle + V(x,y)\right\}.$$

We assume that the mirror step can be easily computed.

2.2 Primal Problem

In this subsection, we consider problem (1) in terms of convergence of the objective function in confidence region. Formally speaking, a vector $\hat{x}$ is called an $(\varepsilon_f, \varepsilon_g, \sigma)$-solution to the primal problem (1), if

$$\begin{aligned} &f(\hat{x}) - f(x^*) \leq \varepsilon_f, \\ &\quad g^{(l)}(\hat{x}) \leq \varepsilon_g\ \forall l \in \{1,\dots,m\} \text{ w.p.} \geq 1-\sigma, \end{aligned} \tag{2}$$

where x^* is a true minimizer of the problem (1). We assume that an algorithm have access only to stochastic subgradient oracles of functions $f, g^{(l)}$ and to δ-approximations $g_\delta^{(l)}$ of constraint functions.

Algorithm 1: Stochastic Mirror Descent with noisy constraints

```
Input: accuracy $\varepsilon > 0$, number of steps $N$,
       stepsize $\eta = \varepsilon/M^2$
1  $x^0 = \text{argmin}_{x\in Q}\, d(x)$;
2  $I = \emptyset, J = \emptyset$;
3  for  $k = 0,1,2,\ldots,N-1$ do
4      if $g_\delta^{(l)}(x^k) \le \varepsilon + \delta \;\forall l \in [m]$ then
5          $x^{k+1} = \text{Mirr}(x^k, \eta\nabla f(x^k, \xi^k))$ ;
            // "productive" steps
6          Add $k$ to $I$
7      else
8          $l(k) = \underset{l\in[m]}{\text{argmax}}\, g_\delta^{(l)}(x^k)$;
9          $x^{k+1} = \text{Mirr}(x^k, \eta\nabla g^{(l(k))}(x^k, \xi^k_{(l(k))}))$ ;
            // "non-productive" steps
10         Add $k$ to $J$;
11 return $\hat{x} = \frac{1}{|I|}\sum_{k\in I} x^k$;
```

Denote $\hat{\nabla}_k f = \nabla f(x^k, \xi^k), \nabla_k f = \nabla f(x^k)$ and $\hat{\nabla}_k g^{(l)} = \nabla g^{(l)}(x^k, \xi^k_{(l)}), \nabla_k g^{(l)} = \nabla g^{(l)}(x^k)$. Additionally, we define $\Theta_0^2 = d(x^*) - d(x^0)$. In these terms we could provide the main theorem.

Theorem 1 *Algorithm 1 with a constant stepsize $\eta = \varepsilon/M^2$ outputs $(\varepsilon, \varepsilon + 2\delta, \sigma)$-solution for any $\varepsilon > 0, \sigma \in (0,1), \delta \ge 0$ in sense of (2) after*

$$N \ge N_0 = \frac{280 \cdot \Theta_0^2 M^2 \log(1/\sigma)}{\varepsilon^2}.$$

The proof of this theorem is given in supplementary material.

Remark 1. Notice that from theoretical point of view selection of the maximum in line 9 of Algorithm 1 could not be avoided. However, in case of non-stochastic constraint computation one of used heuristic is to set $l(k)$ to *any* index of violated constraint and we suggest that such heuristic could work in the noisy setting too.

Remark 2. Notice that a quantity Θ_0^2 is not used in the pseudocode of Algorithm 1. Thus, it is possible to use another initial x^0 and have a warm start such that our algorithm converges faster. This warm start could be obtained by running an algorithm several times with a decreasing value of ε.

Remark 3. We used a constant stepsize for the sake of simplicity. It is possible to adapt techniques of Bayandina et al. (2018); Stonyakin et al. (2019) to use adaptive stepsizes that does not rely on knowledge of Lipschitz constant M.

2.3 Primal-Dual Convergence

In this subsection, we extend properties of the previous algorithm and prove its primal-duality. First of all, let us define the (Lagrange) dual optimization problem associated with the problem (1)

$$\max_{\lambda\in\mathbb{R}^m_+} \left\{ \phi(\lambda) := \min_{x\in Q}\{f(x) + \sum_{i=1}^{n} \lambda_l g^{(l)}(x)\} \right\}. \tag{3}$$

Call λ^* a solution to this dual problem (if it exists). We refer to (Boyd and Vandenberghe, 2004) for an additional background and examples.

It is well-known that for any $x \in Q : g^{(l)}(x) \le 0 \;\forall l \in \{1,\ldots,m\}$ and $\lambda \in \mathbb{R}^m_+$ *the weak duality* holds: $\Delta(x,\lambda) = f(x) - \phi(\lambda) \ge 0$, where Δ is so-called *the duality gap*. We assume that for our primal problem (1) the Slater's condition holds, i.e. $\exists x \in Q : \forall l \in \{1,\ldots,m\} : g^{(l)}(x) < 0$. It implies that the dual problem has a solution and there is *the strong duality*: $\Delta(x^*, \lambda^*) = 0$ for any x^* and λ^* are solutions to the primal and the dual problems respectively.

It gives us a natural way to measure a quality of the pair $(\hat{x}, \hat{\lambda})$ by the value of the duality gap $\Delta(\hat{x}, \hat{\lambda})$. Let us call the pair $(\hat{x}, \hat{\lambda})$ a primal-dual $(\varepsilon_\Delta, \varepsilon_g, \sigma)$-solution to (1) if the following holds with probability at least $1 - \sigma$

$$\begin{aligned} \Delta(\hat{x}, \hat{\lambda}) &\le \varepsilon_\Delta, \\ g^{(l)}(\hat{x}) &\le \varepsilon_g \quad \forall l \in [m]. \end{aligned} \tag{4}$$

Notice that since $\hat{x}$ is not a feasible solution to the primal problem (1), we do not have the weak duality inequality $\Delta(\hat{x}, \hat{\lambda}) \ge 0$. However, the value of duality gap could be controlled from below because of controlled unfeasibility.

The most powerful property of Algorithm 1 is a possibility to generate a pair of primal-dual solutions in sense of (4): we could control the value of the duality gap without the explicit access to the constraint functions.

Following (Nesterov and Shpirko, 2014; Bayandina et al., 2018), we choose the following $\hat{\lambda} \in \mathbb{R}^m_+$ as an estimate of dual variables

$$\hat{\lambda}_l = \frac{1}{|I|}\sum_{k\in J} \mathbb{I}\{l = l(k)\} \tag{5}$$

in terms of Algorithm 1. Additionally, we define useful constant $\overline{\Theta}_0^2 = \sup_{y\in Q}(d(y) - d(x^0))$.

Using $\hat{\lambda}$, we could provide primal-dual properties of Algorithm 1.

Theorem 2 *Let us choose $\hat{\lambda} \in \mathbb{R}^m_+$ as defined in (5) and $\hat{x}$ is an output of Algorithm 1 with a constant stepsize $\eta = \varepsilon/M^2$. Then the pair $(\hat{x}, \hat{\lambda})$ is an $(\varepsilon, \varepsilon + 2\delta, \sigma)$-solution in sense of (4) for any $\varepsilon > 0, \delta \geq 0, \sigma \in (0, 1/2)$ after*

$$N \geq N_0' = \frac{128\overline{\Theta}_0^2 M^2(17\log(2/\sigma) + 2\kappa(E^*))}{\varepsilon^2},$$

where $\kappa(E^*)$ is a constant of Nemirovski's inequality (Boucheron et al., 2013) for the dual space.

Remark 1. If E has a finite dimension d, then we always have $\kappa(E^*) \leq d$. Additionally, if E is endowed with ℓ_p norm, then E^* is endowed with ℓ_q norm, where $1/p + 1/q = 1$, and there is a more precise bound, according to (Dümbgen et al., 2010)

$$\kappa(E^*) \leq K\left(\frac{p}{p-1}, d\right) = \begin{cases} d^{\frac{2}{p}-1}, & p \in [1,2] \\ d^{1-\frac{2}{p}}, & p \in (2, +\infty] \end{cases}$$

In particular, if E has ℓ_2 norm, $\kappa(E^*) = 1$. For $p \in [2, +\infty]$ this bound is tight, however, in the case $p \in [1,2]$ and $d \geq 3$ it could be improved (Boucheron et al., 2013) to, for instance, a logarithmic bound $\kappa(E^*) \leq 2e\log(d) - e$, that could be useful in the case of ℓ_1-norm and an entropy prox-function.

We can write bound on N_0' using O-notation as follows

$$N = O\left(\frac{\overline{\Theta}_0^2 M^2(\log(1/\sigma) + \kappa(E^*))}{\varepsilon^2}\right).$$

The only difference between primal and dual case is connected to the constant in Nemirovski's inequality.

Remark 2. As in the primal case, in the complexity bounds we have a constant $\overline{\Theta}_0^2$ that does not appear in the algorithm description. This fact gives us a chance to work much better in practice than using worst-case constant. The same situation with constant $\kappa(E^*)$.

3 MIXING AMDP

In this section, we discuss the application of the developed algorithm to the problem of approximate solving mixing average-reward Markov Decision Processes (MDP). Firstly, we propose basic definitions connected to MDPs. Next, we discuss some technical nuances that will appear in the algorithm and, finally, we outline the complete algorithm and its parallel implementation.

3.1 Markov Decision Process

An instance of *MDP* is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r})$, where $\mathcal{S}$ is a *finite* set of states; $\mathcal{A} = \bigsqcup_{i \in \mathcal{S}} \mathcal{A}_i$ is a finite state of actions, each set $\mathcal{A}_i$ contains actions from the state i. $\mathcal{P}$ is the collection of state-to-state transition probabilities given actions: $\mathcal{P} = \{p_{ij}(a_i) \mid i,j \in \mathcal{S}, a_i \in \mathcal{A}\}$ where $p_{ij}(a_i)$ is a probability of transition from a state i to a state j given an action a_i. Also, we define $\mathbf{r} \in [0,1]^{|\mathcal{A}|}$ as the state-action reward vector, r_{i,a_i} is the instant reward received when taking the action a_i at the state $i \in \mathcal{S}$. For consistency of notation with work of Jin and Sidford (2020), let $(i,a_i) \in \mathcal{A}$ denote an action a_i at a state i. $A_{\text{tot}} = |\mathcal{A}| = \sum_{i \in \mathcal{S}} |\mathcal{A}_i|$ denotes the total number of state-action pairs. Also we denote by $\mathbf{P}$ the action-state transition probability matrix of size $A_{\text{tot}} \times |\mathcal{S}|$, where $\mathbf{P}_{(i,a_i),j} = p_{ij}(a_i)$ in terms of $\mathcal{P}$.

The goal is to find a stationary (randomized) policy that specifies actions to choose in the fixed state. Formally, a policy π is a block vector such that i-th block corresponds to a probability distribution over $\mathcal{A}_i$. Define as $\mathbf{P}^\pi, \mathbf{r}^\pi$ the transition matrix and the cost vector under the fixed policy π.

Now we are going to define optimality of the policy. In this paper we consider the infinite-horizon *average-reward* MDP with the following objective to maximize

$$\bar{v}^\pi = \lim_{T \to \infty} \mathbb{E}_\pi\left[\frac{1}{T}\sum_{t=1}^T \mathbf{r}_{i_t, a_t} \mid i_1 \sim \mathbf{q}\right].$$

Here $\{i_1, a_1, \ldots, i_t, a_t\}$ are state-actions transitions generated by MDP under a policy π, $\mathbf{q}$ is an initial distribution, and an expectation $\mathbb{E}_\pi[\cdot]$ is taken over trajectories. In our case we interested in the case then the Markov chain generated by an AMDP under a fixed policy π has a unique stationary distribution $\nu^\pi : \nu^\pi \cdot \mathbf{P}^\pi = \nu^\pi$. Notice that in this case the value of $\bar{v}^\pi$ does not depend on an initial distribution $\mathbf{q}$. Then the objective simplifies a lot:

$$\bar{v}^\pi = \langle \nu^\pi, \mathbf{r}^\pi \rangle.$$

Next, we define Bellman equations for an AMDP (Bertsekas, 2005): $\bar{v}^*$ is the optimal average reward if and only if there exists a vector $h^* \in \mathbb{R}^{|\mathcal{S}|}$ satisfying the following

$$\bar{v}^* + h_i^* = \max_{a_i \in \mathcal{A}_i}\left\{\sum_{j \in \mathcal{S}} p_{ij}(a_i) h_j^* + r_{i,a_i}\right\}, \forall i \in \mathcal{S}.$$

We focus on study of the primal LP which solution is equivalent to the solution to Bellman equation

$$\begin{aligned}&\min_{\bar v, h} \bar v\\ &\text{s.t. } \bar v\mathbf{1} + (\hat{\mathbf{I}} - \mathbf{P})h - \mathbf{r} \geq 0,\end{aligned} \tag{6}$$

where $\hat{\mathbf{I}}_{(i,a_i),j} = \mathbf{I}_{i,j}$.

However, without additional assumptions it is hard to analyze problem. Following (Jin and Sidford, 2020; Wang, 2017), we introduce one important assumption on an AMDP instance.

Assumption 1 (Mixing AMDP) *We call an AMDP instance mixing if its so-called mixing time (defined below) is bounded*

$$t_{mix} := \max_\pi \left[\operatorname*{argmin}_{t\geq 1}\left\{\max_{\mathbf{q}} \|(\mathbf{P}^{\pi\top})^t\mathbf{q} - \nu^\pi\|_1 \leq \frac{1}{2}\right\}\right].$$

The most powerful corollary of this result is a possibility to make the search space a compact convex set. Formally speaking, in (Jin and Sidford, 2020) it was proven that the search space for primal variables could be reduced to $\mathcal{X} = [0,1] \times \mathbb{B}_{2R}^{|\mathcal{S}|} = [0,1] \times [-2R,2R]^{|\mathcal{S}|}$, where $R = 2t_{\text{mix}}$. We choose bounds of size $2R$ instead of R because of the same reasons as (Jin and Sidford, 2020) that will be described in Section 3.3.

Overall, we have a (linear) optimization problem on a compact with a large number of constraints

$$\begin{aligned}&\min_{\bar v, h\in\mathcal{X}} \bar v\\ &\text{s.t. } \bar v\mathbf{1} + (\hat{\mathbf{I}} - \mathbf{P})h - \mathbf{r} \geq 0.\end{aligned} \tag{7}$$

If we knew the matrix $\mathbf{P}$, we could apply Stochastic Mirror Descent with constraints (Bayandina et al., 2018) and get an approximate solution to this LP problem. However, it is not the case: we have only sampling access to the transition probability matrix. Another problem we face is computing an optimal policy by an approximate solution to this linear program. It is known that there is a strong connection between the optimal policy and the dual LP but not the primal one. Therefore, we use primal-duality of Algorithm 1 and construct a policy using dual variables.

3.2 Preprocessing

In this subsection we aim to describe complexity of the preprocessing connected to the estimate of the transition probability matrix. We take the estimate of the form

$$\widetilde{\mathbf{P}}_{(i,a_i)} = \frac{1}{n}\sum_{j=1}^n e_{X_j},$$

where X_j are sampled from categorical distribution $\mathbf{P}_{(i,a_i)}$. Choosing appropriate N and compute these quantities for each state-action pair in parallel, we obtain the following proposition.

Proposition 1 *For each δ',$\sigma' > 0$, the estimate $\widetilde{\mathbf{P}}$ of $\mathbf{P}$, such that for each $a \in \mathcal{A}, h \in \mathbb{B}_{2R}^{|\mathcal{S}|} : |\langle \mathbf{P}_{(a)} - \widetilde{\mathbf{P}}_{(a)}, h\rangle| \leq \delta'$ with probability at least $1-\sigma'$, could be computed in $O\left(t_{mix}^2 A_{tot} \cdot \frac{|\mathcal{S}|+\log(A_{tot}/\sigma')}{\delta'^2}\right)$ total samples, $O(1)$ parallel depth and $O\left(t_{mix}^2 \frac{|\mathcal{S}|+\log(A_{tot}/\sigma')}{\delta'^2}\right)$ samples proceed by each single node. In the case of $m \leq A_{tot}$ available workers, it works in $O\left(\frac{A_{tot}}{m} \cdot t_{mix}^2 \frac{|\mathcal{S}|+\log(A_{tot}/\sigma')}{\delta'^2}\right)$ real time.*

The proof of this proposition is given in supplementary.

3.3 Rounding to Optimal Policy

In this subsection, we prove the result that give us a possibility to obtain an approximate optimal policy from the dual variables produced by (1).

Proposition 2 *Suppose that primal $(\bar v^\varepsilon, h^\varepsilon)$ and dual μ^ε variables are $(\varepsilon_f, \varepsilon_g, \sigma)$ -approximate solution to (7) in terms of (expected) primal-dual convergence (4). Define the policy π: $\pi_{i,a_i} = \frac{\mu_{i,a_i}}{\lambda_i}$, where $\lambda \in \mathbb{R}_+^{|\mathcal{S}|}$ is defined as $\lambda_i = \sum_{a_i\in\mathcal{A}_i}\mu_{i,a_i}$. Then π is an $4(\varepsilon_f+\varepsilon_g)$-optimal policy with probability at least $1-\sigma$.*

The proof is given in supplementary material, and it is very similar to the proof of (Jin and Sidford, 2020). There are two differences. Firstly, we have guarantees on our primal-dual solution, not the solution to a saddle-point problem, and this slightly changes the structure of the proof. Secondly, we have high-probability bounds instead of bounds in expectation.

3.4 Parallel Algorithm

In this subsection, we describe a final algorithm to approximate solving AMDP in parallel. In our setup, each single node of a centralized network corresponds to a single state-action pair.

First of all, we describe linear program corresponding to AMDP (7) in terms of (1),

$$\begin{aligned}&\min_{\bar v, h\in\mathcal{X}} f(\bar v, h) = \bar v,\\ &\text{s.t. } g^{(i,a_i)}(\bar v, h) = r_{(i,a_i)} - \bar v\\ &\qquad + (\langle \mathbf{P}_{(i,a_i)}, h\rangle - h_i) \leq 0 \quad \forall (i,a_i) \in \mathcal{A}.\end{aligned} \tag{8}$$

where $\mathcal{X} = [0,1] \times \mathbb{B}_{4t_{\text{mix}}}^{|\mathcal{S}|}$. We set standard Euclidean prox-structure on $\mathcal{X}$: ℓ_2-norm $\|\cdot\|_2$ and a prox-function $d(x) = \frac{1}{2}\|\cdot\|_2^2$. In these terms, we have the following constants: $M = 2$ and $\overline{\Theta}_0^2 = (4R)^2|\mathcal{S}|+1 = O(t_{\text{mix}}^2|\mathcal{S}|)$.

However, in our case we cannot compute constraints since there is no access to the true model. To overcome this, we run preprocessing described in Section 3.2 to derive approximate model $\widetilde{\mathbf{P}}$ and obtain δ-approximation of constraints $g_\delta^{(i,a_i)}$

$$g_\delta^{(i,a_i)}(\bar{v},h) = r_{(i,a_i)} - \bar{v} + (\langle \widetilde{\mathbf{P}}_{(i,a_i)}, h\rangle - h_i). \tag{9}$$

Additionally, we are going to use stochastic subgradients for g by using samples of next state $s \sim \mathbf{P}_{(i,a_i)}$:

$$\nabla_{\bar{v}} g^{(i,a_i)}(\bar{v},h) = -1, \quad \hat{\nabla}_h g^{(i,a_i)}(\bar{v},h,s) = e_s - e_i, \tag{10}$$

with constant $M = 2$. In this case, we can use Algorithm 1 with the following update rules for primal variables $\bar{v}^k$ and h^k. For productive steps ($k \in I$)

$$\bar{v}^{k+1} = \bar{v}^k - \eta, \quad h^{k+1} = h^k, \tag{11}$$

and for non-productive steps ($k \in J$)

$$\bar{v}^{k+1} = \bar{v}^k + \eta, \quad h^{k+1} = h^k - \eta(e_s - e_i), \tag{12}$$

where (i,a_i) is a state-action pair where $\max_{a\in\mathcal{A}} g_\delta^a(\bar{v}^k, h^k)$ attains, and $s \sim \mathbf{P}_{(i,a_i)}$ is a transition sample. We note that in this form this algorithm is sequential.

To design a parallel version of Mirror Descent that presented in Algorithm 2, we are going to use a separate node for each state-action pair (j,a_j). The aim of node corresponds to state-action pair (j,a_j) is 1) compute and update value of $c^k_{(j,a_j)} := g_\delta^{(j,a_j)}$ and 2) sample transitions of MDP.

To compute $c^k_{(j,a_j)}$ faster than in $O(|\mathcal{S}|)$ operations, we note that updates of $\bar{v}^k$ and h^k are sparse and we can store the previous value $c^{k-1}_{(j,a_j)}$.

In the case of productive steps ($k \in I$) we update as follows:

$$c^k_{(j,a_j)} = c^{k-1}_{(j,a_j)} + \eta. \tag{13}$$

This update rule is correct by the update rule (11) and equation (9).

In the case of non-productive steps ($k \in J$) we have a more complicated update

$$\begin{aligned} c^k_{(j,a_j)} = c^{k-1}_{(j,a_j)} - \eta(1 + \widetilde{\mathbf{P}}_{(j,a_j),s} - \widetilde{\mathbf{P}}_{(j,a_j),i} \\ + \mathbb{I}\{j = i\} - \mathbb{I}\{j = s\}). \end{aligned} \tag{14}$$

The correctness of the procedure is guaranteed by the update rule (12) and definition (9). Update rules (13) and (14) gives us an opportunity to update constraints in $O(1)$ time per each node.

Theorem 3 *Let $\varepsilon > 0$ and $\sigma \in (0,1/2)$. The policy $\hat{\pi}$ generated by Algorithm 2 with a constant stepsize $\eta = \varepsilon/64$ and preprocessing described in Section 3.2 performed with parameters $\delta' = \varepsilon/16, \sigma' = \sigma/2$ is an ε-approximate optimal policy with probability at least $1-\sigma$ if*

$$N = O\left(\frac{t^2_{mix}|\mathcal{S}|\log(1/\sigma)}{\varepsilon^2}\right).$$

The described algorithm has $O(1)$ parallel depth, $O(A_{tot}{\cdot}N)$ sample and running time complexity. In the case of $m \le A_{tot}$ available workers, algorithm works in $O\left(\frac{A_{tot}}{m} \cdot t^2_{mix}|\mathcal{S}|\log(1/\sigma)\cdot\varepsilon^{-2}\right)$ real time. The full description is presented in Algorithm 2.

Remark 1. Notice that complexity of preprocessing and Algorithm 2 matches up to logarithmic factors.

Remark 2. Messages "Productive step" and "Non-productive step" ensure that nodes update their values to actual ones.

Remark 3. The communication costs on each round of communication are low: each text message could be send using $O(1)$ bits, and each message with a sample could be sent using only $O(\log|\mathcal{S}|)$ bits.

Remark 4. Additional advantage of the algorithm is a sparsity of updates: there are at most 2 values in the vector h updated each iteration. From the point of view of external memory algorithms, it gives us a possibility to make only 2 requests to the memory if the state-space is too large to store a vector h in RAM.

Remark 5. We use a very coarse bound on $\overline{\Theta}^2 \sim |\mathcal{S}|$. In practice we might expect that $\overline{\Theta}^2 \sim \text{poly}\log|\mathcal{S}|$ and, thus, reduce our dependence on $|\mathcal{S}|$ to logarithmic.

Proof. By Proposition 2, we can produce ε-optimal policy with probability at least $1-\sigma/2$ by running Algorithm 1 on problem (8) with an accuracy $\varepsilon' = \varepsilon/8$ because $4(\varepsilon_\Delta + \varepsilon_g) = 4(\varepsilon' + 2\delta) = 4\varepsilon' + 8\delta = 8\varepsilon'$. Performing union bound for probability of failure for a preprocessing and an algorithm, we have required probability of success of whole scheme.

Since from the point of view of the head node Algorithm 2 is essentially Algorithm 1, we have needed guarantees on number of Mirror Descent iterations by computed constants $M = O(1), \overline{\Theta}_0^2 = O(t^2_{\text{mix}}|\mathcal{S}|), \kappa(E^*) = 1$ and Theorem 2. Notice that the parallel depth of the algorithm is equal to 1.

Total running time complexity forms by additional A_{tot} computations on constraints on each iterations and each computation spends $O(1)$ time by using previous values of computed constraints. The last observation connected to the fact that update of $\bar{v}^k$ and $\bar{h}^k$ also spends $O(1)$ time by sparsity. $\square$

Algorithm 2: Parallel average-reward MDP

```
Input: accuracy $\varepsilon > 0$, number of steps $N$,
       approximation accuracy $\delta = \varepsilon/16$,
       Mirror Descent accuracy $\tilde{\varepsilon} = \varepsilon/16$,
       stepsize $\eta = \tilde{\varepsilon}/4$, confidence level $\sigma$.
Procedure MirrorDescent():
    $\bar{v}^0 = 0; h^0 = 0;$
    for $k = 0,1,2\dots, N-1$ do
        Send "Check constraints" to all nodes;
        Receive $g_\delta^{(i,a_i)}(\bar{v}^k, h^k)\ \forall (i,a_i) \in \mathcal{A}$;
        if $\max_{a \in \mathcal{A}} g_\delta^{(i,a_i)}(\bar{v}^k, h^k) \le \tilde{\varepsilon} + \delta$ then
            $\bar{v}^{k+1} = \bar{v}^k - \eta, \;\; h^{k+1} = h^k$;
            Send "Productive step" to all nodes;
            Add $k$ to $I$;
        end
        else
            $(i,a_i)_k = \mathrm{argmax}_{a \in \mathcal{A}}\, g_\delta^{(i,a_i)}(\bar{v}^k, h^k)$;
            Send "Sample" to node $(i,a_i)_k$;
            Receive state $s$;
            $\bar{v}^{k+1} = \bar{v}^k + \eta$,
             $h^{k+1} = h^k - \eta \cdot (e_s - e_i)$;
            Send "Non-productive step", $i$ and
             $s$ to all nodes;
            Add $k$ to $J$;
        end
    end
    $\hat{\mu}_{(i,a_i)} = \frac{1}{|I|} \sum_{k \in J} \mathbb{I}\{(i,a_i)_k = (i,a_i)\}$;
    return $\pi_{i,a_i} = \mu_{i,a_i} / \left(\sum_{a_i \in \mathcal{A}_i} \mu_{i,a_i}\right)$;
Procedure WorkerNode($j, a_j$):
    Compute $\widetilde{\mathbf{P}}_{(j,a_j)}$ with precision $\delta$ and
     confidence level $\sigma/2$;
    $c^0_{(j,a_j)} = r_{j,a_j}, k = 0$;
    while is not finished do
        Wait message;
        if "Compute constraints" then
            Send $c^k_{(j,a_j)}$ as $g_\delta^{(j,a_j)}(\bar{v}^k, h^k)$;
        end
        else if "Sample" then
            Sample $s \sim \mathbf{P}_{(j,a_j)}$ and send it;
        end
        else if "Productive step" then
            $k = k + 1$;
            $c^k_{(j,a_j)} = c^{k-1}_{(j,a_j)} + \eta$;
        end
        else if "Non-productive step" then
            Receive $s$ and $i$;
            $k = k + 1$;
            $c^k_{(j,a_j)} = c^{k-1}_{(j,a_j)} - \eta(1 + \widetilde{\mathbf{P}}_{(j,a_j),s} -$
             $\widetilde{\mathbf{P}}_{(j,a_j),i} + \mathbb{I}\{j = i\} - \mathbb{I}\{j = s\})$;
        end
    end
```

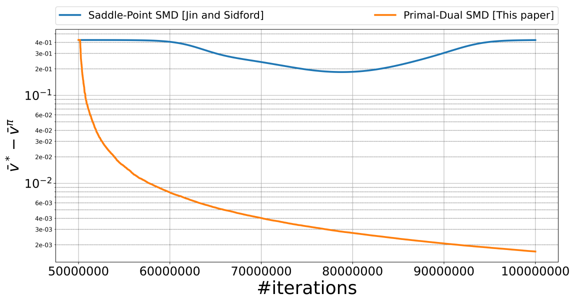

Figure 1: Comparison between Stochastic Mirror Descent (Jin and Sidford, 2020) and Algorithm 2 on RiverSwim environment with 6 states and 2 actions.

4 NUMERICAL EXPERIMENTS

In this section we perform numerical comparison of Algorithm 2 in *sequential setting* with algorithm described in paper (Jin and Sidford, 2020).

At first, we highlight technical features. In both mentioned algorithms, the value of mixing time t_{mix} is needed. However, computation of maximum over policies of discrete-valued function seems to be a very hard problem to be computed precisely. We replace the value of t_{mix} by its estimate $\widehat{t_{\text{mix}}}$ computed over 1000 randomly generated policies. Additionally, to make precise comparison we compute optimal average-reward value v^* using LP-representation of the problem (6) for known model, and the average-reward value v^π of the given policy π by computing a stationary distribution using known model.

For our comparison, we apply algorithms to two different environments: RiverSwim (Strehl and Littman, 2008), Access-Control Queuing task (Sutton and Barto, 2018). We choose these environments because we have an exact model for them, and they require non-trivial exploration.

RiverSwim. We start with the environment description. RiverSwim is an environment with 6 states in a row with 2 actions for each state: swim to the left or swim to the right. Swimming to the left is always successful. For the first state, this action returns to itself and gives 0.005 reward. Swimming to the right move the agent to the right state with probability 0.35, make the agent stay in the current state with probability 0.6, and move to the left with probability 0.05. For the last state moving to the right state returns the agent to this last state and gives 1 reward. Thus, the optimal average-reward policy is always swimming to the right.

Estimate value of $\widehat{t_{\text{mix}}}$ is equal to 155 and it makes this environment hard for our algorithm 2 and Stochastic

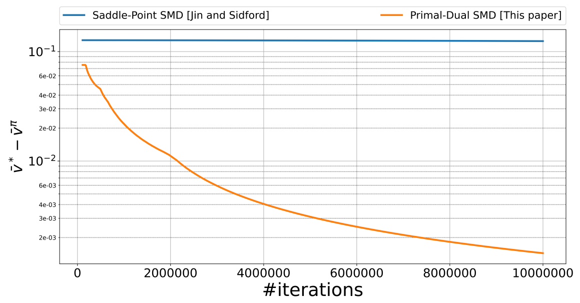

Figure 2: Comparison between Stochastic Mirror Descent (Jin and Sidford, 2020) and Algorithm 2 on Access-Control Queuing task with 10 servers.

Mirror Descent (SMD) (Jin and Sidford, 2020). We compare these algorithm with $\varepsilon = 10^{-2}$ and use 1000 samples for preprocessing, the result is presented on Figure 1. Significant superior of our approach could be explained by independence of stepsize of Algorithm 2 to the value of t_{mix}, whereas stepsize in algorithm of (Jin and Sidford, 2020) depends on this quanitity as $1/t_{\text{mix}}^2$. In the case of this MDP, stepsize of SMD is smaller in $2.5 \cdot 10^4$ times. Additionally, notice that we generated a very little number of samples during preprocessing, and therefore these computation does not affect the total running time. Additionally, high sparsity of rewards explains the high value of the objective function during first iterates.

Access-Control Queuing task. This environment is modelling very practical situation for using average-reward MDP model. For the description we refer to (Sutton and Barto, 2018). The only difference is normalization of rewards to make them in the interval [0,1].

Estimated value of mixing time $\widehat{t_{\text{mix}}} = 44$ and has dense rewards. On Figure 2 we present comparison between Algorithm 2 and SMD with $\varepsilon = 10^{-2}$ and 500 samples for preprocessing. Again, since the value of t_{mix} is relatively large for this MDP, worse convergence of SMD is explained by smaller stepsize that is required by theoretical analysis. Our algorithm is almost agnostic to the value of t_{mix} and it makes it much more effective for large values of t_{mix}.

5 CONCLUSION

In this work, we proposed a parallel algorithm for solving an average-reward MDP. As far as we know, it is the first parallel algorithm in the generative model setting without reduction to the discounted problem. The interesting properties of the provided method are very low communication costs between the head node and all other nodes and the sparsity of updates that offer a possibility to work with a very large state space.

Another contribution is the development of Mirror Descent with constraints algorithms. We provide an algorithm that works with inexact computation of constraints and prove its primal-dual properties. The setting of inexact computation of constraints was developed in (Lan and Zhou, 2020) but results on primal-dual convergence of such algorithms appeared in known literature only in a deterministic exact case (Bayandina et al., 2018).

Turning to possible extensions, there arise natural questions.

Could a preprocessing step be avoided and make the algorithm model-free? In the current version, we needed to do a required number of preprocessing iterations to guarantee the condition on constraints. It seems possible to use an unbiased stochastic oracle for constraint evaluation.

Another question is connected to the total work complexity. The cost of a high level of parallelism is worse total running time and sample complexity in comparison to algorithms based on saddle-point formulations. It is connected to two theoretical issues: 1) ℓ_1-approximation of the model and 2) performing Mirror Descent with box constraints. The first issue could be resolved using another approximation metric that respects MDP structure as it was done in (Agarwal et al., 2020). The second issue is fundamental for non-linear optimization due to existing lower bounds (Guzmán and Nemirovski, 2015). The only way to avoid it is to use the linear structure of mixing AMDP problem. Could the total work time and sample complexity be reduced without losing the possibility to run in parallel?

The last question is about further generalizations of our approach. One of the interesting directions is to use our algorithm for solving constrained MDP as it is done in the paper (Jin and Sidford, 2020). However, we note that it is possible to use non-linear constraints in our case.

Acknowledgements

The publication was supported by the grant for research centers in the field of AI provided by the Analytical Center for the Government of the Russian Federation (ACRF) in accordance with the agreement on the provision of subsidies (identifier of the agreement 000000D730321P5Q0002) and the agreement with HSE University No. 70-2021-00139.

References

Agarwal, A., Kakade, S., and Yang, L. F. (2020). Model-based reinforcement learning with a generative model is minimax optimal. In *Conference on Learning Theory*, pages 67–83. PMLR.

Agrawal, S. and Jia, R. (2017). Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Azar, M. G., Munos, R., and Kappen, B. (2012). On the Sample Complexity of Reinforcement Learning with a Generative Model. Appears in Proceedings of the 29th International Conference on Machine Learning (ICML 2012).

Bartlett, P. L. and Tewari, A. (2009). Regal: A regularization based algorithm for reinforcement learning in weakly communicating mdps. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, page 35–42, Arlington, Virginia, USA. AUAI Press.

Bayandina, A., Dvurechensky, P., Gasnikov, A., Stonyakin, F., and Titov, A. (2018). *Mirror Descent and Convex Optimization Problems with Non-smooth Inequality Constraints*, pages 181–213. Springer International Publishing, Cham.

Bertsekas, D. (2005). *Dynamic Programming and Optimal Control*. Number 1 in Athena Scientific optimization and computation series. Athena Scientific.

Boucheron, S., Lugosi, G., and Massart, P. (2013). *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. NY Cambridge University Press.

Dümbgen, L., Geer, S., Veraar, M., and Wellner, J. (2010). Nemirovski's inequalities revisited. *The American mathematical monthly : the official journal of the Mathematical Association of America*, 117:138–160.

Guzmán, C. and Nemirovski, A. (2015). On lower complexity bounds for large-scale smooth convex optimization. *J. Complex.*, 31:1–14.

Jaksch, T., Ortner, R., and Auer, P. (2010). Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(51):1563–1600.

Jin, Y. and Sidford, A. (2020). Efficiently solving MDPs with stochastic mirror descent. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4890–4900. PMLR.

Jin, Y. and Sidford, A. (2021). Towards tight bounds on the sample complexity of average-reward mdps. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5055–5064. PMLR.

Kearns, M. and Singh, S. (2002). Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2):209–232.

Lan, G., Nemirovski, A., and Shapiro, A. (2012). Validation analysis of mirror descent stochastic approximation method. *Mathematical programming*, 134(2):425–458.

Lan, G. and Zhou, Z. (2020). Algorithms for stochastic optimization with expectation constraints. *Computational Optimization and Applications*, 76.

Li, G., Wei, Y., Chi, Y., Gu, Y., and Chen, Y. (2020). Breaking the sample size barrier in model-based reinforcement learning with a generative model. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12861–12872. Curran Associates, Inc.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609.

Nemirovski, A. and Yudin, D. (1983). *Problem Complexity and Method Efficiency in Optimization*. A Wiley-Interscience publication. Wiley.

Nesterov, Y. (2009). Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1):221–259. First appeared in 2005 as CORE discussion paper 2005/67.

Nesterov, Y. and Shpirko, S. (2014). Primal-dual subgradient method for huge-scale linear conic problems. *SIAM Journal on Optimization*, 24(3):1444–1457.

Sidford, A., Wang, M., Wu, X., Yang, L. F., and Ye, Y. (2018a). Near-optimal time and sample complexities for solving markov decision processes with a generative model. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5192–5202.

Sidford, A., Wang, M., Wu, X., and Ye, Y. (2018b). Variance reduced value iteration and faster algorithms for solving markov decision processes. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 770–787. SIAM.

Stonyakin, F., Alkousa, M., Stepanov, A., and Tytov, A. (2019). Adaptive mirror descent algorithms for convex and strongly convex optimization problems with functional constraints. *Journal of Applied and Industrial Mathematics*, 13:557–574.

Strehl, A. L. and Littman, M. L. (2008). An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331. Learning Theory 2005.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

Szepesvari, C. (2010). *Algorithms for Reinforcement Learning*. Morgan and Claypool Publishers.

Wang, M. (2017). Primal-dual π learning: Sample complexity and sublinear run time for ergodic markov decision problems.

Supplementary Material: Primal-Dual Stochastic Mirror Descent for MDPs

A MISSING PROOFS

A.1 Proof of Theorem 1

To simplify the notation, we introduce the stochastic gradient noise function

$$\gamma_k(y) = \begin{cases} \eta\langle \hat{\nabla}_k f - \nabla_k f, y - x^k\rangle, & k \in I; \\ \eta\langle \hat{\nabla}_k g^{(l(k))} - \nabla_k g^{(l(k))}, y - x^k\rangle, & k \in J. \end{cases} \tag{15}$$

The main property of this quantity is that it forms a martingale-difference sequence.

Now we are going to provide some useful properties of Algorithm 1 in terms of bounds of error in objective function f and constraint satisfiability. Afterwards, we combine all properties to a convergence analysis of the algorithm. But at first, we state one useful technical result. We will actively use the following lemma:

Lemma 1 (Bayandina et al. (2018)) *Let h be some convex function over a set Q, $\eta > 0$ is a stepsize, $x \in Q$. Let the point $x^+ = \mathrm{Mirr}(x, \eta(\nabla h(x) + \Delta))$, where Δ is some vector from the dual space. Then, for any $y \in Q$*

$$\eta(h(x) - h(y) + \langle \Delta, x - y\rangle) \le \eta\langle \nabla h(x) + \Delta, x - y\rangle$$
$$\le \frac{\eta^2}{2}\|\nabla h(x) + \Delta\|_*^2 + V(x,y) - V(x^+, y).$$

Lemma 2 *For a point $\hat{x}$ produced by Algorithm 1 and any $y \in Q$ the following holds*

$$\eta|I|\cdot(f(\hat{x}) - f(y)) \le \frac{\eta^2 M^2}{2}N + [d(y) - d(x^0)] - |J|\eta\cdot\varepsilon$$
$$+ \sum_{k=0}^{N-1}\gamma_k(y) + \eta\sum_{k\in J} g^{(l(k))}(y). \tag{16}$$

Proof. By the construction of "productive" and "non-productive" steps in Algorithm 1 and Lemma 1, we have for all $y \in Q$

$$\eta(f(x^k) - f(y)) \le \frac{\eta^2 M^2}{2} + V(x^k, y) - V(x^{k+1}, y) + \eta\langle \hat{\nabla}_k f - \nabla_k f, y - x^k\rangle, \qquad k \in I;$$

$$\eta(g^{(l(k))}(x^k) - g^{(l(k))}(y)) \le \frac{\eta^2 M^2}{2} + V(x^k, y) - V(x^{k+1}, y) + \eta\langle \hat{\nabla}_k g^{(l(k))} - \nabla_k g^{(l(k))}, y - x^k\rangle \qquad k \in J.$$

By definition of δ-approximation, we have the following inequalities for "productive" and "non-productive" steps respectively

$$\eta(f(x^k) - f(y)) \le \frac{\eta^2 M^2}{2} + V(x^k, y) - V(x^{k+1}, y) + \gamma_k(y),$$
$$\eta(g_\delta^{(l(k))}(x^k) - g^{(l(k))}(y)) \le \frac{\eta^2 M^2}{2} + V(x^k, y) - V(x^{k+1}, y) + \gamma_k(y) + \eta\delta.$$

Sum all these inequalities over all $k \in I$ and $k \in J$ and use the fact that $I \cup J = \{0, \dots, N-1\}$

$$\begin{aligned}&\sum_{k\in I} \eta(f(x^k) - f(y)) + \sum_{k \in J} \eta(g_\delta^{(l(k))}(x^k) - g^{(l(k))}(y)) \\ &\le \frac{\eta^2 M^2}{2}|I| + \frac{\eta^2 M^2}{2}|J| + \sum_{k=0}^{N-1}[V(x^k,y) - V(x^{k+1},y)] + \sum_{k=0}^{N-1}\gamma_k(y) + |J|\eta\delta.\end{aligned} \tag{17}$$

By the choice of $x^0 = \operatorname{argmin}_{x\in Q} d(x)$, we have

$$\sum_{k=0}^{N-1}[V(x^k,y) - V(x^{k+1},y)] \le V(x^0,y) = d(y) - d(x^0) - \langle \nabla d(x^0), y - x^0\rangle = d(y) - d(x^0).$$

Using definition of "non-productive" steps $g_\delta^{(l(k))}(x^k) > \varepsilon + \delta$ and convexity of f

$$\begin{aligned}&\sum_{k\in I} \eta(f(x^k) - f(y)) + \sum_{k \in J} \eta(g_\delta^{(l(k))}(x^k) - g^{(l(k))}(y)) \\ &\ge \eta|I|(f(\hat{x}) - f(y)) + \eta|J|(\varepsilon + \delta) - \eta\sum_{k\in J} g^{(l(k))}(y).\end{aligned}$$

By application of inequality (17) and simple regrouping of terms, we finish the proof. □

During this section we set $y = x^*$. In this case $d(x^*) - d(x^0) \le \Theta_0^2$. Further we are going to derive bound on $g^{(l)}$ at $\hat{x}$ for each separate function $g^{(l)}$:

Lemma 3 *For $\hat{x}$ produced by Algorithm 1 the following holds*

$$g^{(l)}(\hat{x}) \le \varepsilon + 2\delta. \tag{18}$$

Proof. By convexity and definition of a δ-approximation

$$g^{(l)}(\hat{x}) \le \frac{1}{|I|}\sum_{k\in I} g^{(l)}(x^k) \le \frac{1}{|I|}\sum_{k\in I}(g_\delta^{(l)}(x^k) + \delta).$$

We finish the proof by definition of "productive" steps and a set I. □

Before obtaining final bounds on f we prove some technical fact:

Lemma 4 *For any $y \in Q$: $\|y - x^0\|^2 \le 2(d(y) - d(x^0))$.*

Proof. Follows from strongly convexity of d with respect to norm $\|\cdot\|$ and the fact that x^0 is a minimum of d. □

Now we derive bounds on sums of our stochastic noise function with high probability.

Lemma 5 *Define event $\mathcal{E}$ such that the following inequalities holds*

$$\sum_{k=0}^{N-1}\gamma_k(x^*) < \frac{2\Theta_0}{M}\cdot\sqrt{2N\varepsilon^2\log(1/\sigma)}.$$

Then $\Pr[\mathcal{E}] \ge 1 - \sigma$.

Proof. Define filtration of σ-algebras $\mathcal{F}_k = \sigma\left(\{\xi^i, \xi^i_{(l)}\}_{i\le k}\right)$. Notice that $\gamma_k(x^*)$ is a martingale-difference sequence adapted to $\mathcal{F}_k$.

Let us derive bound on the sum. Notice that by Holder inequality and Lemma 4

$$|\gamma_k(x^*)| \le 2\eta M\|x^* - x^k\| \le \frac{4\varepsilon\Theta_0}{M},$$

Then by Azuma-Hoeffding inequality

$$\Pr\left[\sum_{k=0}^{N-1} \gamma_k(x^*) \geq t_1\right] \leq \exp\left(\frac{-t_1^2}{2N \cdot (16\varepsilon^2\Theta_0^2)/(M^2)}\right);$$

By setting $t_1 = 4\Theta_0 M^{-1} \cdot \sqrt{2N\varepsilon^2 \log(1/\sigma)}$ we finish the proof. □

One could apply Lemma 5 to inequalities in Lemmas 2 and 3:

Corollary 1 *Under event $\mathcal{E}$ defined in Lemma 5, the following inequalities holds for all $l \in [m]$*

$$\eta|I|(f(\hat{x}) - f(x^*)) < \eta|I|\varepsilon - \frac{\varepsilon^2 N}{2M^2} + \Theta_0^2 + \frac{4\Theta_0\sqrt{2N\varepsilon^2\log(1/\sigma)}}{M};$$
$$g^{(l)}(\hat{x}) < \varepsilon + 2\delta.$$

Proof. The second inequality follows directly from Lemma 5. The first one uses definition $\eta = \varepsilon/M^2$, inequalities $g^{(l))}(x^*) \leq 0$ and, finally, Lemma 5. □

Now we are going to prove last technical result before the statement of the main theorem of this subsection:

Lemma 6 *If $\sigma < 1$ and*

$$N \geq N_0 = \frac{280 \cdot \Theta_0^2 M^2 \log(1/\sigma)}{\varepsilon^2}$$

then

$$\frac{\varepsilon^2 N}{2M^2} - \Theta_0^2 - \sqrt{2N\varepsilon^2}\left(\frac{4\Theta_0\sqrt{\log(1/\sigma)}}{M}\right) \geq 0.$$

Proof. Fix variable $t = 2N\varepsilon^2$. Then we have quadratic inequality of type $at^2 - (b_1+b_2)t - c \geq 0$. By exact formula for quadrativ equation and Taylor approximation we know that inequality holds if $t > (b_1+b_2)/a + c/(b_1+b_2)$. By numeric inequality $1/(b_1+b_2) \leq 1/b_1$ for nonnegative b_1,b_2, we have that enough to choose $t \geq b_1/a + b_2/a + c/b_1$. Using our choice of N and numeric inequality $2a^2 + 2b^2 \geq (a+b)^2$

$$\sqrt{2N\varepsilon^2} \geq \sqrt{280} \cdot \Theta_0 M\sqrt{\log(1/\sigma)} \geq 16\Theta_0\sqrt{\log(3/\sigma)} \cdot M + \frac{\Theta_0^2 M}{2\Theta_0\sqrt{\log(1/\sigma)}}.$$

Here we use that $\log(1/\sigma) \geq 1$ and $16 + 1/2 \leq \sqrt{280}$. □

Finally, we are ready to state theorem that describe convergence properties of Algorithm 1.

Proof of Theorem 1. To guarantee that $f(\hat{x}) - f(x^*) \leq \varepsilon$ with probability at least $1-\sigma$, we use the first inequality in Corollary 1 and Lemma 6. To guarantee satisfaction of constraints, we simply use Corollary 1. □

A.2 Proof of Theorem 2

Recall our estimate of a dual variables

$$\hat{\lambda}_l = \frac{1}{|I|}\sum_{k\in J} I\{l = l(k)\}.$$

Lemma 7 *Suppose $\hat{x}$ is an output of Algorithm 1 and $\hat{\lambda}$ is defined as in (5) and $\overline{\Theta}_0^2 \geq \sup_{y\in Q}(d(y) - d(x^0))$. Then the following holds*

$$\eta|I|\Delta(\hat{x},\hat{\lambda}) \leq \frac{\eta^2 M^2}{2}N + \overline{\Theta}_0^2 - |J|\eta\varepsilon + \sum_{k=0}^{N-1}\gamma_k(x^0) + \overline{\Theta}_0\sqrt{2}\cdot\left\|\sum_{k=0}^{N-1}\Delta_k\right\|_*, \tag{19}$$

where Δ_k is defined as follows

$$\Delta_k = \begin{cases} \eta\left(\hat{\nabla}_k f - \nabla_k f\right), & k \in I \\ \eta\left(\hat{\nabla}_k g^{(l(k))} - \nabla_k g^{(l(k))}\right), & k \in J. \end{cases}$$

Proof. We start from Lemma 2. Here we move all terms consist of y to the right-hand side and minimize over y

$$\eta|I|f(\hat{x}) \le \frac{\eta^2 M^2}{2} N - |J|\eta\varepsilon$$
$$+ \min_{y \in Q}\left\{ d(y) - d(x^0) + \sum_{k=0}^{N-1} \gamma_k(y) + \eta|I|f(y) + \eta \sum_{k\in J} g^{(l(k))}(y) \right\}.$$

Notice that by definition of $\hat{\lambda}$ we have

$$\eta|I|\phi(\hat{\lambda}) = \min_{y\in Q}\left\{ \eta|I|f(y) + \eta\sum_{k\in J} g^{(l(k))} \right\}.$$

Thus, we have to upper bound $d(y) - d(x^0)$ and $\sum_{k=0}^{N-1}\gamma_k(y)$ without dependence on y to obtain required result. The first upper bound is trivial: $d(y) - d(x^0) \le \overline{\Theta}_0^2$ by definition of $\overline{\Theta}_0^2$.

To analyse the second term, we use the definition of γ_k in terms of Δ_k and Holder inequality

$$\sum_{k=0}^{N-1}\gamma_k(y) = \sum_{k=0}^{N-1}\langle \Delta_k, y - x^0 + x^0 - x^k\rangle \le \|y - x^0\| \left\| \sum_{k=0}^{N-1}\Delta_k \right\|_* + \sum_{k=0}^{N-1}\gamma_k(x^0).$$

The last step is to apply Lemma 4 and definition of $\overline{\Theta}_0^2$ to obtain uniform bound on $\|y - x^0\|$. □

Our next goal is to derive bound on the right-hand side of (19). It is possible using concentration of measure techniques as in Lemma 5.

Lemma 8 *Define event $\mathcal{E}'$ such that the following inequalities holds*

$$\sum_{k=0}^{N-1}\gamma_k(x^0) < \frac{2\overline{\Theta}_0}{M}\cdot\sqrt{2N\varepsilon^2\log(2/\sigma)}$$
$$\left\|\sum_{k=0}^{N-1}\Delta_k\right\|_* < \frac{\sqrt{2\kappa(E^*)} + \sqrt{4\log(2/\sigma)}}{M}\sqrt{2N\varepsilon^2},$$

where Δ_k is defined in 7 and $\kappa(E^*)$ is a constant of Nemirovski's inequality (Boucheron et al., 2013) for the dual space. Then $\Pr[\mathcal{E}'] \ge 1 - \sigma$.

Remark. If E has a finite dimension d, then we always have $\kappa(E^*) \le d$. Additionally, if E is endowed with ℓ_p norm, then E^* is endowed with ℓ_q norm, where $1/p + 1/q = 1$, and there is a more precise bound, according to (Dümbgen et al., 2010)

$$\kappa(E^*) \le K\left(\frac{p}{p-1}, d\right) = \begin{cases} d^{\frac{2}{p}-1}, & p \in [1,2] \\ d^{1-\frac{2}{p}}, & p \in (2, +\infty] \end{cases}$$

In particular, if E has ℓ_2 norm, $\kappa(E^*) = 1$. For $p \in [2, +\infty]$ this bound is tight, however, in the case $p \in [1,2]$ and $d \ge 3$ it could be improved to, for instance, a logarithmic bound $\kappa(E^*) \le 2e\log(d) - e$, that could be useful in the case of ℓ_1-norm and an entropy prox-function. *Proof.* The case of first inequality is identical to Lemma 5 by rescaling σ to $\sigma/2$.

To ensure the last inequality, we apply bounded difference inequality (Boucheron et al., 2013). This follows by observing that $Z = \|\sum_{k=0}^{N-1}\Delta_k\|_*$ satisfies bounded difference condition with a constant $4\cdot\varepsilon\underline{M}^{-1}$ that is greater than $2\cdot\|\Delta_k\|_*$ a.s.. Thus

$$\Pr[Z - \mathbb{E}Z > t_2] \le \exp\left(\frac{-t_2^2}{2\cdot N\varepsilon^2\cdot 4M^{-2}}\right).$$

Take $t_2 = 2M^{-1}\sqrt{2N\varepsilon^2} \cdot \sqrt{\log(4/\sigma)}$ and the last we have to do is to bound expectation of Z. Here we apply Nemirovski's inequality

$$\left(\mathbb{E}\left\|\sum_{k=0}^{N-1} \Delta_k\right\|_*\right)^2 \le \mathbb{E}\left[\left\|\sum_{k=0}^{N-1} \Delta_k\right\|_*^2\right] \le \kappa(E^*)\sum_{k=0}^{N-1}\mathbb{E}\left[\|\Delta_k\|_*^2\right] \le \kappa(E^*)\frac{4N\varepsilon^2}{M^2}.$$

By application of the union bound we finish the proof. □

Corollary 2 *Under event $\mathcal{E}'$ defined in Lemma 8, the following inequalities holds for all $l \in [m]$*

$$\eta|I|\Delta(\hat{x},\hat{\lambda}) < \eta|I|\varepsilon - \frac{\varepsilon^2 N}{2M^2} + \overline{\Theta}_0^2 + \sqrt{2N\varepsilon^2}\left(\frac{2\overline{\Theta}_0(4\sqrt{\log(2/\sigma)} + \sqrt{2\kappa(E^*)})}{M}\right);$$

$$g^{(l)}(\hat{x}) < \varepsilon + 2\delta.$$

Proof. Inequalities on $g^{(l)}(\hat{x})$ is equivalent to the same in 1. Inequality on $\Delta(\hat{x},\hat{\lambda})$ follows from a combination of Lemma 7 and Lemma 8. □

Now we are ready to state a technical lemma that is similar to Lemma 6.

Lemma 9 *If $\sigma \le 1/2$ and*

$$N \ge N_0' = \frac{128\overline{\Theta}_0^2 M^2(17\log(2/\sigma) + 2\kappa(E^*))}{\varepsilon^2}$$

then

$$\frac{\varepsilon^2 N}{2M^2} - \overline{\Theta}_0^2 - \sqrt{2N\varepsilon^2}\left(\frac{2\overline{\Theta}_0(4\sqrt{\log(2/\sigma)} + \sqrt{2\kappa(E^*)})}{M}\right) \ge 0.$$

Proof. Using the same reasoning about solving quadratic inequality in terms of $t = \sqrt{2N\varepsilon^2}$ as in Lemma 6 it is sufficient to show that

$$\sqrt{2N\varepsilon^2} \ge 4\left(2\overline{\Theta}_0 M(4\sqrt{\log(2/\sigma)} + \sqrt{2\kappa(E^*)})\right)$$
$$+ \frac{\overline{\Theta}_0^2 M}{2\overline{\Theta}_0(4\sqrt{\log(2/\sigma)} + \sqrt{2\kappa(E^*)})}.$$

Since $\sigma \le 1/2 \Rightarrow \log(2/\sigma) \ge 1$ and $\sqrt{2\kappa(E^*)} \ge 1$, we have that it is sufficient to show that

$$\sqrt{2N\varepsilon^2} \ge 4\left(2\overline{\Theta}_0 M((4 + 1/20)\sqrt{\log(2/\sigma)} + \sqrt{2\kappa(E^*)})\right).$$

By numeric inequality $2a^2 + 2b^2 \ge (a+b)^2$ and $(4 + 1/20)^2 \le 17$, it is satisfied with our choice of $N \ge N_0'$. □

Now we are ready to prove our main result.

Proof of Theorem 2. The inequality on $g(\hat{x})$ is satisfies by Corollary 2. We have to satisfy inequality on duality gap. The inequality on duality gap follows directly from Corollary 2 and Lemma 9 since $N \ge N_0'$. Notice that a probability $1 - \sigma$ appears from the event $\mathcal{E}'$ defined in Lemma 8. □

A.3 Proof of Proposition 1

At first, we prove one technical lemma.

Lemma 10 (Estimation of parameters of categorical distribution) *Suppose that we have samples $\{X_i\}_{i=1}^N, X_i \in \{1,\ldots,d\}$ drawn from categorical distribution with a parameter $s \in \Delta^d$. Define an empirical estimate $\hat{s} = \frac{1}{N}\sum_{i=1}^N e_{X_i} \in \Delta^d$, where e_j is an j-th standard basis vector.*

Then, for any δ',$\sigma' > 0$, the inequality $\|s - \hat{s}\|_1 \le \delta'$ holds with probability at least $1 - \sigma'$ if

$$N \ge \frac{8d + 4\log(1/\sigma')}{\delta'^2}.$$

Proof. First of all, notice that $\mathbb{E} e_{X_i} = s$. Denote by Δ a centred random variable $\hat{s} - s = \frac{1}{N}\sum_{i=1}^n (e_{X_i} - s)$. Then, by Nemirovski's inequality and bound on the ℓ_1 norm of elements of a simplex, we might estimate the mean of the square of ℓ_1 norm of this random variable: $\mathbb{E}\|\Delta\|_1^2 \leq 4d/N$.

Let us define function $f(X_1, \ldots, X_N) = \|\Delta\|_1$. It might be checked that this function satisfies conditions of the bounded difference inequality (Boucheron et al., 2013) with constant $2/N$. Thus, for all $t > 2\sqrt{d/N} \geq \mathbb{E}\|\Delta\|_1$

$$\Pr[\|\Delta\|_1 > t] = \Pr[\|\Delta\|_1 - \mathbb{E}\|\Delta\|_1 > t - \mathbb{E}\|\Delta\|_1] \leq \exp\left(\frac{-(t - 2\sqrt{d/N})^2}{2N^{-1}}\right).$$

Taking $N \geq (8d + 4\log(1/\sigma')) \cdot (\delta')^{-2}$ and $t = \delta'$, we finish the proof. □

Using this simple lemma, we can easily obtain required sample complexity for the preprocessing even in parallel sampling setting. Remember that we are going to use Algorithm 1, hence, we want to approximate each constraint function.

Proof of Proposition 1. Notice that each $\mathbf{P}_{(a)} \in \Delta^{|\mathcal{S}|}$ is a parameters of a categorical distribution we sampling from, and the estimator from the previous lemma could be applied.

By Holder's inequality and the definition of the search space for h we have that $|\langle \mathbf{P}_{(a)} - \widetilde{\mathbf{P}}_{(a)}, h\rangle| \leq 2M \cdot \|\mathbf{P}_{(a)} - \widetilde{\mathbf{P}}_{(a)}\|_1$. Hence, to make it less than δ', we required to make ℓ_1 norm of difference less than $\delta'/(2M)$. To make all conditions work simultaneously, it is enough to make the probability in the terms of the previous lemma less than σ'/A_{tot} and apply the union bound over all A_{tot} conditions. The last observation that finishes the proof is that sampling for an estimation of each $\widetilde{\mathbf{P}}_a$ could be done separately and independent. □

A.4 Proof of Proposition 2

Proof. Conditions of the proposition give us the following guarantees in terms of the duality gap and constraint satisfaction with needed probability $\geq 1 - \sigma$

$$\begin{aligned} &\bar{v}^\varepsilon - \min_{\bar{v}, h}\left(\bar{v} + (\mu^\varepsilon)^\top\left(-\bar{v}\mathbf{1} - (\hat{\mathbf{I}} - \mathbf{P})h + \mathbf{r}\right)\right) \leq \varepsilon_f, \\ &\bar{v}^\varepsilon \mathbf{1} + (\hat{\mathbf{I}} - \mathbf{P})h^\varepsilon - \mathbf{r} \geq -\varepsilon_g \mathbf{1}. \end{aligned} \tag{20}$$

We can rewrite the first condition in more suitable terms of λ^ε

$$\forall \bar{v}, \forall h \in \mathcal{X} : \varepsilon_f \geq (\bar{v}^\varepsilon - \bar{v}) + \bar{v} \cdot (\lambda^\varepsilon)^\top \mathbf{1} + (\lambda^\varepsilon)^\top([I - \mathbf{P}^\pi]h - \mathbf{r}^\pi). \tag{21}$$

Now we have all required instruments and we can bound the expectation of an average value of our policy

$$v^\pi = (\nu^\pi)^\top \mathbf{r}^\pi = (\nu^\pi - \lambda^\varepsilon)^\top([\mathbf{P}^\pi - I]h^\varepsilon + \mathbf{r}^\pi) + (\lambda^\varepsilon)^\top([\mathbf{P}^\pi - I]h^\varepsilon + \mathbf{r}^\pi).$$

Here we used the stationary of our policy: $(\nu^\pi)^\top(\mathbf{P}^\pi - I) = 0$. For simplicity, we remind that $\mu^\varepsilon, \lambda^\varepsilon \geq 0$, and, hence, $\langle \mu^\varepsilon, \mathbf{1}\rangle = \langle \lambda^\varepsilon, \mathbf{1}\rangle = \|\lambda^\varepsilon\|_1$.

Firstly, bound the second term by 21

$$(\lambda^\varepsilon)^\top([\mathbf{P}^\pi - I]h^\varepsilon + \mathbf{r}^\pi) \geq \bar{v}^\varepsilon - \bar{v}(1 - \|\lambda^\varepsilon\|_1) - \varepsilon_f.$$

To bound the first term we will use Lemma 7 from (Jin and Sidford, 2020):

$$\|(\mathbf{I} - \mathbf{P}^\pi + \mathbf{1}(\nu^\pi)^\top)^{-1}\|_\infty \leq M.$$

Also we have for all $\mu \geq 0$ by primal feasibility: $\mu^\top([\hat{\mathbf{I}} - \mathbf{P}]h^* - \mathbf{r} + \bar{v}^*\mathbf{1}) \geq 0$. We can combine it with our condition (21) for arbitrary h and $\bar{v} = \bar{v}^*$, using the fact that $\bar{v}^\varepsilon \geq \bar{v}^* - \varepsilon_g$

$$\begin{aligned} \varepsilon_f + \varepsilon_g &\geq \bar{v}^* - \bar{v}^* + (\mu^\varepsilon)^\top([\hat{\mathbf{I}} - \mathbf{P}]h - \mathbf{r} + \bar{v}^*\mathbf{1}) \\ &\geq (\mu^\varepsilon)^\top([\hat{\mathbf{I}} - \mathbf{P}]h - \mathbf{r} + \bar{v}^*\mathbf{1}) - (\mu^\varepsilon)^\top([\hat{\mathbf{I}} - \mathbf{P}]h^* - \mathbf{r} + \bar{v}^*\mathbf{1}) \\ &= (\mu^\varepsilon)^\top([\hat{\mathbf{I}} - \mathbf{P}](h - h^*)) = (\lambda^\varepsilon)^\top[\mathbf{I} - \mathbf{P}^\pi](h - h^*). \end{aligned}$$

Hence, we have

$$\begin{aligned} 2M\|(\lambda^{\varepsilon})^{\top}[\mathbf{I}-\mathbf{P}^{\pi}]\|_1 &= \max_{h\in\mathbb{B}_{2M}^{|\mathcal{S}|}} (\lambda^{\varepsilon})^{\top}[\mathbf{I}-\mathbf{P}^{\pi}]h \\ &= \max_{h\in\mathbb{B}_{2M}^{|\mathcal{S}|}} (\lambda^{\varepsilon})^{\top}[\mathbf{I}-\mathbf{P}^{\pi}](h-h^*) + (\lambda^{\varepsilon})^{\top}[\mathbf{I}-\mathbf{P}^{\pi}]h^* \\ &\le \varepsilon_f + \varepsilon_g + M\|(\lambda^{\varepsilon})^{\top}[\mathbf{I}-\mathbf{P}^{\pi}]\|_1. \end{aligned}$$

Combining with the fact that $\nu^{\pi}(\mathbf{I}-\mathbf{P}^{\pi}) = 0$, we obtain$\|(\nu^{\pi}-\lambda^{\varepsilon})^{\top}[\mathbf{I}-\mathbf{P}^{\pi}]\|_1 \le \frac{\varepsilon_f+\varepsilon_g}{M}$. By almost the same argument as in (Jin and Sidford, 2020), we also have $|\langle \nu^{\pi}-\lambda^{\varepsilon}, \mathbf{r}^{\pi}\rangle| \le \varepsilon_f+\varepsilon_g$. Hence, there is a bound on the required first term:

$$(\nu^{\pi}-\lambda^{\varepsilon})^{\top}([\mathbf{P}^{\pi}-I]h^{\varepsilon}+\mathbf{r}^{\pi}) \ge -2M\cdot\frac{(\varepsilon_f+\varepsilon_g)}{M} - (\varepsilon_f+\varepsilon_g).$$

Overall, we obtain the required inequality by taking $\bar{v}=0$ and by feasibility of the pair $(\bar{v}^{\varepsilon}+\varepsilon_g, h^{\varepsilon})$

$$\bar{v}^{\pi} \ge \bar{v}^{\varepsilon} - \bar{v}(1-\|\lambda^{\varepsilon}\|_1) - \varepsilon_f - 3(\varepsilon_f+\varepsilon_g) \ge \bar{v}^* - 4(\varepsilon_f+\varepsilon_g).$$

□