An online convex optimization algorithm for controlling linear systems with state and input constraints

Marko Nonhoff and Matthias A. Müller

Abstract—This paper studies the problem of controlling linear dynamical systems subject to point-wise-in-time constraints. We present an algorithm similar to online gradient descent, that can handle time-varying and a priori unknown convex cost functions while restraining the system states and inputs to polytopic constraint sets. Analysis of the algorithm’s performance, measured by dynamic regret, reveals that sublinear regret is achieved if the variation of the cost functions is sublinear in time. Finally, we present an example to illustrate implementation details as well as the algorithm’s performance and show that the proposed algorithm ensures constraint satisfaction.

I. INTRODUCTION

Application of methods from online learning and optimization leads to new techniques for learning-based controller synthesis. In this paper, we apply the online convex optimization (OCO) framework first introduced in [1] to the problem of controlling constrained linear dynamical systems. OCO is an online variant of classical numerical optimization, where the cost function to be minimized is time-varying and a priori unknown. Specifically, at every time $t$, an algorithm chooses an action $y_t \in \mathbb{Y}$ from a convex constraint set $\mathbb{Y}$ based on the chosen actions in the past and the observed cost functions. After the action $y_t$ is chosen, the environment reveals a new cost function $L_t : \mathbb{Y} \rightarrow \mathbb{R}$ which leads to the cost $L_t(y_t)$. The goal is to minimize the total cost $\sum_{t=1}^{T} L_t(y_t)$ over $T$ stages. This problem has been studied extensively in the online optimization and learning community (see [2], [3] for an overview) with a focus on the non-asymptotic performance of the algorithms. Typically, dynamic regret is chosen as a performance measure, which is defined as the cumulative gap between the cost observed by the algorithm and some benchmark [4]–[6]. A sublinear regret bound is desirable, implying that the algorithm’s performance is asymptotically on average no worse than the benchmark. This framework enjoys advantages in its ability to handle time-varying unknown cost functions while ensuring constraint satisfaction and the low computational complexity of its algorithms, which are desirable in controller synthesis, too.

In the classical OCO framework, one typically considers that any action $y_t \in \mathbb{Y}$ can be chosen. Thereby, no underlying dynamical system can be considered. Application of OCO to the control of dynamical systems has already been studied by introducing a switching cost or ramp cost $d(y_t - y_{t-1})$ to study the effect of a time coupled cost function [7]. In particular, in [8], a switching cost which can be seen as an additional quadratic cost on the input $u_t - u_{t-1}$ of a single integrator system $y_t = y_{t-1} + u_{t-1}$ is studied, which is then extended to the case of general linear systems in [9]. A similar approach is taken in [10], where a quadratic regret bound for general controllable linear systems is derived. In [11]–[13], linear dynamical systems subject to quadratic cost functions are considered and it is shown, that the regret with respect to the best linear controller is sublinear. Therein, the algorithms update a linear control strategy at every time step. This approach is extended in [14] to general convex cost functions. Whereas in the classical OCO framework the allowed actions $y_t$ are typically restricted to a constraint set $\mathbb{Y}$, none of these previous works on the combination of OCO and dynamical systems considers state or input constraints. A similar setting is considered in [15]–[17], where online optimization is employed to control the output of an exponentially stable system to the steady-state solutions of an optimization problem. While constraints on the solutions of the optimization problems can be considered, point-wise-in-time constraints on both the input and state trajectories of the dynamical system are not. This approach is extended to stabilizable LTI systems in [18].

This work builds on and extends the results in [10] such that constraints on both the inputs and states of the system are guaranteed to be satisfied at all times. Ensuring satisfaction of these constraints is of paramount importance, e.g., in safety-critical applications or where only a finite amount of control energy is available. As discussed above, ensuring constraint satisfaction is not possible in the existing results in the literature and it requires substantial modifications in the algorithm design and the corresponding analysis. Our analysis reveals that the proposed algorithm still enjoys sublinear regret, as is the case for existing algorithms which cannot guarantee satisfaction of input and state constraints.

This paper is organized as follows. Section II defines the problem setting, whereas our algorithm is proposed and discussed in Section III. We proceed to give a regret analysis and our main theorem in Section IV. In Section V, we illustrate implementation details and the performance of the proposed algorithm. Section VI concludes the paper.

Notation: For a vector $x \in \mathbb{R}^n$, $x_i$ denotes its $i$-th entry and $\|x\|$ the Euclidean norm, whereas for a matrix $A \in \mathbb{R}^{n \times m}$, $A_i$ is its $i$-th row and $\|A\|$ the corresponding induced matrix norm. Given a set $S \subseteq \mathbb{R}^n$, $\Pi_S(x) = \min_{s \in S} \|x - s\|^2$ is the projection of $x \in \mathbb{R}^n$ onto the set $S$. We define by $\mathbb{N}_{[a,b]}$ the set of natural numbers in the interval $[a,b]$. The gradient of a function $f(x)$ evaluated at $x$ is denoted by

©2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
∇f(x). Additionally, I_n is the identity matrix of size n × n.

II. SETTING

We consider discrete-time linear systems of the form

\[ x_{t+1} = Ax_t + Bu_t, \quad x(0) = x_0, \]

where \( x_t \in \mathbb{R}^n \) are the states of the system and \( u_t \in \mathbb{R}^m \) are the control inputs. The matrices \( A \in \mathbb{R}^{n \times n} \) and \( B \in \mathbb{R}^{n \times m} \) are assumed to be known. System (1) is subject to state constraints \( x_t \in \mathcal{X} \) and input constraints \( u_t \in \mathcal{U} \) which have to be satisfied at every time instant \( t \in \mathbb{N}_{[1,T]} \). We assume both constraint sets \( \mathcal{X} \) and \( \mathcal{U} \) to be compact polytopes.

Assumption 1: The state and input constraint sets are compact convex polytopes with 0 in their interior, given by \( \mathcal{X} = \{ x \in \mathbb{R}^n | C_x x \leq d_x \} \) and \( \mathcal{U} = \{ u \in \mathbb{R}^m | C_u u \leq d_u \} \), where \( C_x \in \mathbb{R}^{c_x \times n}, d_x \in \mathbb{R}^{c_x}, C_u \in \mathbb{R}^{c_u \times m}, \) and \( d_u \in \mathbb{R}^{c_u} \).

Note that compactness of \( \mathcal{U} \) implies existence of a finite constant \( D_u \) such that \( \| u_1 - u_2 \| \leq D_u \) for all \( u_1, u_2 \in \mathcal{U} \).

The online control problem under consideration in this work is described as follows: At every time step \( t \in \mathbb{N}_{[1,T]} \), the current state \( x_t \) is measured. The controller has to decide a control input \( u_t \in \mathcal{U} \), based only on the measured state and previous cost functions, and apply the input \( u_t \) to system (1). Afterwards, the cost function \( L_t : \mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R} \) is revealed by the environment resulting in the cost \( L_t(x_t, u_t) \). Finally, the system evolves to the next state \( x_{t+1} \). The goal is to minimize the total cost over \( T \) stages.

As common in OCO, we consider regret as a measure for our algorithm's performance. The optimal state and input sequence \( x^*_t = \{ x^*_1, \ldots, x^*_T \} \) and \( u^*_t = \{ u^*_1, \ldots, u^*_T \} \), respectively, are the solution to the optimization problem

\[
\min_{x \in \mathcal{X}^T, u \in \mathcal{U}^T} \sum_{t=0}^{T} L_t(x_t, u_t) \quad \text{s.t.} \quad x_{t+1} = Ax_t + Bu_t.
\]

Thus, \((x^*_t, u^*_t)\) denote the optimal states and inputs at time instant \( t \) in hindsight, when all cost functions \( L_t \) are known. Then, in our case, we define the dynamic regret \( R \) as

\[
R = \sum_{t=0}^{T} L_t(x_t, u_t) - L_t(x^*_t, u^*_t).
\]

The dynamic regret \( R \) measures how much performance is lost due to not knowing the cost functions \( L_t \) a priori. The definition in (2) is in line with the dynamic regret measure imposed in [9]. Another popular regret measure is comparing the algorithm's performance to the best linear feedback controller [11]–[14], which is a weaker benchmark since the optimal trajectories may not result from a linear feedback. Next, we require some technical assumptions, which are fairly standard in OCO (compare [6], [8]–[10]).

Assumption 2: For every \( t \in \mathbb{N}_{[0,T]} \), the cost function \( L_t \) satisfies

1) \( L_t(x, u) = f_t^x(x) + f_t^u(u) \),
2) \( f_t^x(x) \) is \( \alpha_x \)-strongly convex, \( l_x \)-smooth for all \( x \in \mathcal{X} \),
3) \( f_t^u(u) \) is \( \alpha_u \)-strongly convex, \( l_u \)-smooth for all \( u \in \mathcal{U} \).

Note that Lipschitz continuity of the cost functions \( f_t^x : \mathcal{X} \rightarrow \mathbb{R} \) and \( f_t^u : \mathcal{U} \rightarrow \mathbb{R} \) with Lipschitz constants \( L_x \) and \( L_u \) follows from \( l \)-smoothness and compactness of the constraint sets \( \mathcal{X} \) and \( \mathcal{U} \), respectively.

Additionally, we define \( \theta_t = \arg \min_{x \in \mathcal{X}} f_t^x(x) \) and \( \eta_t = \arg \min_{u \in \mathcal{U}} f_t^u(u) \). Note that due to compactness of the sets \( \mathcal{X} \) and \( \mathcal{U} \) and strong convexity of the cost functions, the minima are attained, finite, and unique. In contrast to the trajectories \( x^* \) and \( u^* \), the sequences \( \theta_t = \{ \theta_1, \ldots, \theta_T \} \) and \( \eta_t = \{ \eta_1, \ldots, \eta_T \} \) in general do not satisfy the system dynamics (1). If the cost functions \( L_t \) are allowed to change at every time instant, we will not be able to achieve low dynamic regret. Therefore, we define a measure for the variation of the cost functions similar to [6], [8] as Path length: \( \sum_{t=0}^{T} \| \theta_t - \theta_{t-1} \| + \sum_{t=0}^{T} \| \eta_t - \eta_{t-1} \| \).

Next, we require some technical assumptions, which are imposed in [9]. Another popular regret measure is comparing the algorithm's performance to the best linear feedback controller [11]–[14], which is a weaker benchmark since the optimal trajectories may not result from a linear feedback. Next, we require some technical assumptions, which are fairly standard in OCO (compare [6], [8]–[10]).

Assumption 3 states that the minimum \((\theta_t, \eta_t)\) of the cost function \( L_t(x_t, u_t) \) at time instant \( t \) is a feasible steady state with respect to the system dynamics (1) and the constraints.

Assumption 4: The pair \((A, B)\) is controllable, i.e., \( \text{rank}(B) = n + 1 \).

As discussed in [10], a bound on \( \| A \| \), which can be seen as a bound on the instability of system (1), is necessary since we want to control the system by applying one gradient descent step at every time instant \( t \). Therefore, one gradient descent step needs to be able to counteract the instability of the system, which yields Assumption 4. It can also be seen that, if \( \alpha_x = l_x \), which is the case for, e.g., \( f_t^x(x) = \| x - \theta \|^2 \) for some \( \theta \in \mathbb{R}^n \), then any controllable system satisfies Assumption 4. Finally, we require that any state in \( \mathcal{X} \) can be reached from every initial state in \( \mathcal{X} \) in finite time.

Assumption 5: There exists a constant \( \mu \in \mathbb{N} \) such that for every two states \( x, y \in \mathcal{X} \), there exists a feasible input trajectory \( u = \{ u^{(1)}, \ldots, u^{(\mu)} \} \) satisfying \( A^\mu x + S_\mu u = y \), where \( S_\mu = (B \ A B \ldots A^{\mu-1}B) \).

An input trajectory \( u = \{ u_1, \ldots, u_T \} \), \( \tau \in \mathbb{N} \) is called feasible if it satisfies both the input and state constraints, i.e., \( u_t \in \mathcal{U} \) and \( x_t \in \mathcal{X} \) when applying \( u_t \) for all \( t \in \mathbb{N}_{[1,T]} \). Assumption 5 can be interpreted as assuming controllability under constraints. If Assumption 5 is not satisfied for a state constraint set \( \mathcal{X}_i \) and an input constraint set \( \mathcal{U}_i \), a suitable
Algorithm 1 (OGD for constrained linear systems)

Given step sizes $\gamma_t$ and $\gamma_x$, initialization $\hat{u}_0$, $v_0$, and measured state vector $x_{t-1}$.
At time $t \in [1, T]$:

(v) $v_t = \Pi_\epsilon (v_{t-1} - \gamma_v \nabla f_\epsilon (v_{t-1}))$

(3)

(\hat{v}_t) = \{\hat{u}_{t-1}^{(2)}, \ldots, \hat{u}_{t-1}^{(\mu)}, v_t\}$

(4)

(\hat{x}_{t+\mu} = A^\mu x_t + S_c \hat{v}_t \nabla f_\epsilon (\hat{x}_{t+\mu})$

(5)

if $\|z_{t+\mu} - x_{t+\mu}\| = 0$ set $\alpha_t = 0$, else

(7)

$\alpha_t = \frac{1}{1 + \delta_t}$

(8)

Find $g_t \in \mathcal{U}^\mu$ such that

$\nabla x^g (\tau; x_t) \in \mathcal{X} \forall \tau \in \mathbb{N}_{t+\mu}$

(9a)

$g_t (\tau) \in \mathcal{U} \forall \tau \in \mathbb{N}_{t+\mu-1}$

(9b)

$A^\mu x_t + S_c g_t = x_{t+\mu} + \delta_t (x_{t+\mu} - \hat{x}_{t+\mu})$

(9c)

$\hat{u}_t = (1 - \alpha_t) \hat{v}_t + \alpha_t g_t$

(10)

$u_t = \hat{u}_t^{(1)}$

(11)

subset $\mathcal{X}$ of the viability kernel\(^2\) has to be found that renders Assumption 1 and Assumption 5 satisfied.

Remark 6: Whereas Assumption 5 itself is natural in our setting, we assume the constant $\mu$ to be known in Algorithm 1. This potentially leads to a large prediction horizon and degrading performance, see Section III for details. The question how Algorithm 1 needs to be modified in order to shorten the prediction horizon while maintaining a sublinear regret bound is an interesting problem for future research.

III. ALGORITHM

Before we state our algorithm, we first define some useful notation. Given an input sequence $u = \{u^{(1)}, u^{(2)}, \ldots, u^{(\mu)}\}$, where $u^{(i)} \in \mathbb{R}^n$, we denote by $u = ((u^{(i)})^T \ldots (u^{(1)})^T)^T$ the vector created by stacking the components of $u$. Moreover, we write $x^u (\tau; x_t)$ for the state at time $\tau \in [t, t + \mu]$ when starting at $x^u (t; x_t) = x_t$ and applying the sequence $u$.

The proposed OCO scheme is given in Algorithm 1. In our framework described above, at every time instant $t$, Algorithm 1 computes a control input $u_t$ based on the measured state vector $x_t$ and the previous cost function $L_{t-1}$. Then, only after applying the control input $u_t$ to system (1), a new cost function $L_t$ is observed, resulting in the cost $L_t(x_t, u_t)$. Note that the feasibility problem in (9) always has a solution: Since the state $x_{t+\mu} + \delta_t (x_{t+\mu} - \hat{x}_{t+\mu})$ is contained in $\mathcal{X}$ by the definition of $\mathcal{X}$, Assumption 5 states that it can be reached from $x_t \in \mathcal{X}$ in $\mu$ time steps.

Roughly speaking, Algorithm 1 predicts the trajectories of system (1) and then applies online gradient descent (OGD)\(^3\) (11, [3]) twice to track the optimal input $\eta_t$ and the optimal state $\theta_t$. For that, the proposed algorithm can be separated into three steps. First, OGD is applied in (5) to compute an estimate $v_t$ of the optimal input $\eta_t$. Second, OGD is applied again to track the optimal state $\theta_t$ in (4)-(6). Similar to warm-starting in MPC [23], a candidate input sequence $\hat{v}_t$ for the next $\mu$ time steps is generated by shifting the previously predicted input sequence $\hat{u}_t$ and extending it by $v_t$ in (4). This input sequence is then used to predict the state $\mu$ time steps in the future in (5) and OGD is applied again to calculate a desired state $x_{t+\mu}$ improving the state cost in (6). Last, the predicted input sequence $\hat{u}_{t-1}$ is updated in (7)-(10) by computing a feasible input sequence $g_t$ in (9). In contrast to [10], Algorithm 1 employs a convex combination of $\hat{v}_t$ and $g_t$ in (10) instead of committing to applying the updated predicted input sequence for the next $\mu$ time steps. Note that application of the whole predicted sequence $\hat{u}_t$ yields $x^\hat{u}_t (t + \mu; x_t) = x_{t+\mu}$ as shown in (14) in the Appendix. The whole procedure is illustrated in Figure 1.

At every time step $t$, Algorithm 1 solves two projections in (3) and (6). Additionally, a feasibility problem has to be solved in (9) if the predicted state $\hat{x}_{t+\mu}$ is not optimal. The two projections in (3) and (6) are, in general, projections onto convex polytopic sets. In particular, they are computationally cheap if the constraint sets $\mathcal{U}$ and $\mathcal{X}$ have a simple shape, such as, e.g., box constraints. The feasibility problem in (9) can be cast as a linear feasibility program since all constraint sets are polytopes resulting in linear constraints.

IV. REGRET ANALYSIS

In this section, we state our main result, a bound on the regret of Algorithm 1. Its proof is given in the appendix.

Theorem 7: Let $x_0 \in \mathcal{X}$ and Assumptions 1-5 be satisfied. Given a feasible initialization $\hat{u}_0, v_0$, and step sizes $\gamma_t \leq \frac{2}{\gamma_a + \gamma_x}$ and $\frac{\|A\|^{-1}}{\|A\| + 1} < \gamma_x \leq \frac{1}{\gamma_a + \gamma_x}$, the dynamic regret $R$ of Algorithm 1 can be upper bounded by

\[ R \leq \sum_{t=1}^T \left( \frac{\|A\|^{-1}}{\|A\| + 1} \right)^t \]
\[ \mathcal{R} \leq C_0 + C_\varrho \sum_{t=0}^{T} \| \theta_t - \theta_{t-1} \| + C_\eta \sum_{t=0}^{T} \| \eta_t - \eta_{t-1} \| , \]

for some constants \( C_0, C_\varrho, C_\eta > 0 \) independent of \( T \). Moreover, \( x_t \in \mathcal{X} \) and \( u_t \in \mathcal{U} \) for all \( t \in [0, T] \).

Theorem 7 states that the regret of Algorithm 1 is linear in the path length. Hence, we achieve sublinear regret if the path length is sublinear in \( T \). This result is well aligned with other results in the literature, see, e.g., [6], [8], [9]. Despite the presence of input and state constraints in our setting, we achieve the same sublinear regret bound as in the unconstrained case [10] up to constant factors. Note that, as already discussed in [10], this result implies asymptotic convergence to the optimal equilibrium if it holds that \( (\theta_t, \eta_t) = (\theta_{t'}, \eta_{t'}) \) for some \( t' \in \mathbb{N} \) and all \( t \geq t' \). In addition, Theorem 8 guarantees constraint satisfaction for every time instant \( t \in \mathbb{N} [0, T] \).

V. SIMULATIONS

In this section, we illustrate the implementation of Algorithm 1 and its performance and compare it to the algorithm presented in [10], which cannot guarantee constraint satisfaction. We consider the same system as in [10] given by

\[
\begin{pmatrix}
    x_{t+1,1} \\
    x_{t+1,2} \\
    x_{t+1,3}
\end{pmatrix} = \begin{pmatrix}
    1.05 & 0.7 & 1.75 \\
    0.35 & 0.7 & 1.05 \\
    1.4 & 0.105 & 1.855
\end{pmatrix} \begin{pmatrix}
    x_t \\
    u_t \\
    0
\end{pmatrix} + \begin{pmatrix}
    1 \\
    0 \\
    1
\end{pmatrix} u_t
\]

and add state and input constraints
\[
x_t \in \mathcal{X}_0 = \{ x \in \mathbb{R}^3 | |x_1| \leq 3, |x_2| \leq 2, |x_3| \leq 1 \},
\]
\[
u_t \in \mathcal{U} = \{ u \in \mathbb{R} \mid |u| \leq 4 \}.
\]

Unfortunately, the state constraint set \( \mathcal{X}_0 \) does not satisfy Assumption 5 because it includes states which are either not reachable by a feasible input sequence or outside of the viability kernel. We apply methods from [24] and the Multi-Parametric Toolbox [25] to compute a subset \( \tilde{\mathcal{X}} \) of the viability kernel that satisfies Assumption 5 with \( \mu = 6 \).

The shrinked constraint set \( \tilde{\mathcal{X}} \) is calculated as detailed in Section II with \( \delta = 0.01 \). Moreover, we choose cost functions \( L_t(x, u) = f^x_t(x) + f^u_t(u) = \frac{1}{2} ||x - \theta_t||^2 + \frac{1}{2} ||u - \eta_t||^2 \). The optimal state and input \( (\theta_t, \eta_t) \) are time-varying and a priori unknown. The initial condition \( x_0 \) and initial feasible inputs \( v_0, \mathbf{a}_0 \) were all set to 0. The step sizes were chosen as \( \gamma_x = 0.98 \) and \( \gamma_v = 0.98 \) for both algorithms, satisfying the assumptions of Theorem 7. The auxiliary input \( g_t \) in (9) in Algorithm 1 is found by the 'linprog' command in Matlab.

Figure 2 shows the simulated closed loop for both algorithms, the optimal states \( \theta_t \) and inputs \( \eta_t \) and the corresponding constraints. It can be seen that Algorithm 1 tracks the optimal equilibrium. Compared to the algorithm presented in [10], Algorithm 1 is slower to react to setpoint changes. This is due to the fact that controllability under constraints in Assumption 5 requires \( \mu = 6 \), whereas \( \mu = n = 3 \) could be used in [10] (see Remark 6). On the other hand, in contrast to the algorithm from [10], Algorithm 1 is guaranteed to satisfy the constraints at all times.

VI. CONCLUSION

In this work, we apply online convex optimization to linear dynamical systems subject to polytopic state and input constraints. We give an online algorithm that achieves sublinear regret if the variation of the cost functions, measured in path length, is sublinear and guarantees constraint satisfaction.

There are two obvious directions for future research. On the one hand, the prediction horizon could be shortened as discussed in Remark 6 which may result in the algorithm not being able to apply a full gradient step within the shorter horizon. In this case, new analysis techniques are required to prove that a sublinear regret bound still holds. On the other hand, Assumption 3 could be relaxed, allowing economic cost functions.

APPENDIX

Before we prove Theorem 7 we give some auxiliary results. First, since \( \theta_t \) and \( \eta_t \) are only defined for \( 0 \leq t \leq T \), we fix without loss of generality for the remainder of this work \( \eta_t = \mathbf{v}_0 \) and \( \theta_t = \tilde{x}_0 \) for all \( t < 0 \).

Second, in order to shorten notation, let \( \tilde{\alpha}_t = 1 - \alpha_t \) and

\[
\tilde{\alpha}_j = \begin{cases}
    \prod_{i=j}^{t-1} \tilde{\alpha}_s & \text{if } i < j \\
    \tilde{\alpha}_i & \text{if } i = j \\
    1 & \text{if } i > j
\end{cases}.
\]

We have by the definition of \( \alpha_t \) in (8) that \( 0 \leq \alpha_t < 1 \) and, hence, \( 0 < \tilde{\alpha}_t \leq 1 \) as well as \( 0 < \tilde{\alpha}_{t+s} \leq 1 \) for any \( s \in \mathbb{N} \). Moreover, since \( \alpha_t + \tilde{\alpha}_t = 1 \), it holds for any \( \tau, s \in \mathbb{N} \) that

\[
\tilde{\alpha}_{\tau+s} + \sum_{j=0}^{s} \tilde{\alpha}_{\tau+s}^{j+1} \alpha_{\tau+j} = 1. \tag{12}
\]

Next, we have the following result on the rate of convergence of projected gradient descent [19].
and $l$-smooth function $f : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}$ to be minimized, one projected gradient step $x_t = \Pi_{\mathcal{X}}(x_{t-1} - \gamma \nabla f(x_{t-1}))$, where $\gamma \leq \frac{2}{\omega_1}$ is a step size parameter, satisfies
\[
\|x_t - \theta_t\| \leq \kappa \|x_{t-1} - \theta_{t-1}\|, \tag{13}
\]
where $\theta_t = \arg \min_{x \in \mathcal{X}} f(x)$ and $\kappa = 1 - \alpha \gamma$. Accordingly, we define $\kappa_x = 1 - \alpha_x \gamma_x$ and $\kappa_u = 1 - \alpha_u \gamma_u$.

Third, we examine the closed-loop trajectories and the predicted trajectories of Algorithm 1. Using (10) and the definition of $\alpha_t$ in Algorithm 1 it can be shown that
\[
A^\mu_t x_t + S_c \hat{u}_t = x_{t+1}^\pi + \beta \hat{u}_t^{(1)} + B \hat{u}_t + B \eta_t, \tag{14}
\]
holds in both cases, $\alpha_t = 0$ as well as $\alpha_t \neq 0$. Then, the predicted states $\hat{x}_{t+1}$ can be calculated recursively as follows
\[
\hat{x}_{t+1} = A^\mu_t (Ax_t + B \hat{u}_t) + B \eta_t. \tag{15}
\]
Moreover, the predicted input $\hat{u}_t$ can be expressed in terms of previous inputs by
\[
\hat{u}_t = \sum_{j=0}^\infty \alpha_t^{j+1} \alpha_{t-j} \hat{g}_{t-j} \tag{16}
\]
The real state trajectory is $x_{t+1} = A^\mu_t x_t + S_c \hat{u}_t$, where the inputs $u_{t+s}, s \in \mathbb{N}[0,t+1]$, can be expressed by repeatedly inserting (10) by
\[
u_{t+s} = \hat{u}_t^{(1)} + \hat{u}_t^{(2)} \ldots + \hat{u}_t^{(s+2)} + \sum_{j=0}^s \alpha_t^{j+1} \alpha_{t-j} \hat{g}_{t-j} \tag{17}
\]
if $0 \leq s \leq \mu - 1$, and
\[
u_{t+s} = \sum_{j=0}^{\mu-1} \alpha_t^{j+1} \alpha_{t-j} \hat{g}_{t-j} + \hat{u}_t, \tag{18}
\]
if $s = \mu - 1$. Next, we are ready to state the following lemma, which bounds the cumulative prediction error.

**Lemma 8:** Let Assumptions 15 be satisfied. Given step sizes $\gamma_u \leq \frac{2}{\omega_1 + \omega_2}$ and $\|A\| \leq \frac{1}{\omega_1 + \omega_2}$, it holds that
\[
\sum_{t=0}^{T-\mu} \|x_{t+1} - x_t\| \leq \mu C_1 \|\theta_t - \theta_{t-1}\| + \kappa \|B\| \|\eta_t - \eta_{t-1}\| \tag{19}
\]
where $C_1 = \frac{D_u(2(\mu - 1)(\mu - 2))}{\omega_1 + \omega_2}$.

**Proof:** First, note that the step size $\gamma_x$ is well-defined due to the bound on $\|A\|$ in Assumption 4.

Then, by inserting (13) we have
\[
\sum_{t=0}^{T-\mu-1} \|v_{t+1} - \eta_t\| \leq \kappa_u \sum_{t=0}^{T-\mu} \|v_t - \eta_{t-1}\| + \kappa \|B\| \|\eta_t - \eta_{t-1}\|. \tag{20}
\]
Due to $v_0 = \eta_0$ and $1 - \kappa_u > 0$ rearranging yields
\[
\sum_{t=0}^{T-\mu} \|v_{t+1} - \eta_t\| \leq \frac{\kappa_u}{1 - \kappa_u} \sum_{t=0}^{T-\mu} \|\eta_t - \eta_{t-1}\|. \tag{21}
\]
Additionally, since $\theta_{t-1} = \hat{\theta}_t$, we have by Assumption 3
\[
\sum_{t=0}^{T-\mu} \|\hat{x}_{t+1} - \theta_{t-1}\| \leq \sum_{t=0}^{T-\mu} \|\hat{x}_{t+1} - \theta_t\| \tag{22}
\]
and
\[
\sum_{t=0}^{T-\mu} \|A x_{t+1} - B \eta_t - B \eta_{t-1}\| \leq \sum_{t=0}^{T-\mu} \|A x_{t+1} - B \eta_{t+1} - B \eta_{t-1}\| \tag{23}
\]
Moreover, using the triangle inequality we get
\[
\sum_{t=0}^{T-\mu} \|\hat{x}_{t+1} - x_{t+1}\| \leq (1 + \kappa) \sum_{t=0}^{T-\mu} \|\hat{x}_{t+1} - \theta_{t-1}\| \tag{24}
\]
and
\[
\sum_{t=0}^{T-\mu} \|A x_{t+1} - B \eta_{t+1} - B \eta_{t-1}\| \leq \sum_{t=0}^{T-\mu} \|A x_{t+1} - B \eta_{t+1} - B \eta_{t-1}\| \tag{25}
\]
Last, we combine all the above results to prove Lemma 7.

By (5), we get $\hat{x}_{t+1} - x_{t+1} = S_c \left( \hat{u}_t^{(1)} - u_{t+1}^{(1)} \right) + \hat{u}_t^{(2)} - u_t^{(2)} + B \eta_t$, which implies that the right-hand-side of the equation can be rewritten using (17). By adding $S_c \left( \eta_{t-1}^{(1)} \ldots \eta_{t+1}^{(1)} \right)^T - S_c \left( \eta_{t-1}^{(2)} \ldots \eta_{t+1}^{(2)} \right)^T$, taking the norm on both sides, using (22) and (23), we arrive at
\[
\|\hat{x}_{t+1} - x_{t+1}\| \leq \sum_{t=0}^{T-\mu-1} \|v_{t+1} - \eta_{t+1} - \eta_t\| + \mu D_u \sum_{j=0}^{\mu-1} \sum_{j=1}^{\mu-1} \eta_{t-j}
\]
By summing over $t$ on both sides we get
\[
\sum_{t=0}^{T-\mu} \|\hat{x}_{t+1} - x_{t+1}\| \leq \sum_{t=0}^{T-\mu} \|v_{t+1} - \eta_t\| + \|\eta_{t+1} - \eta_t\| \tag{27}
\]
It remains to insert (18) and (20) to get the result.
and then discuss feasibility of the states and inputs. In order to obtain an upper bound for the regret, we begin by bounding the suboptimality of the chosen control inputs. By the definition of $u_t$ in (11) we have

$$
\sum_{t=0}^{T} \| u_t - \eta_t \| = \sum_{t=0}^{T} \| \hat{u}_t - \eta_t \| 
$$

where we throw away terms $0 < \hat{\alpha}_t < 1$ and used a telescoping series and the triangle inequality in the last line. By (18) and the definition of $\hat{\alpha}_t$ in (10), it holds that

$$
\sum_{t=0}^{T} \| u_t - \eta_t \| \leq C_1 \sum_{t=0}^{T} \| \hat{\alpha}_t - \eta_t \| + C_2 \sum_{t=0}^{T} \| \theta_t - \bar{\theta}_t \| ,
$$

where

$$
C_1 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{\hat{u}}_i \| , C_2 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{u}_i \| , C_3 = \left( \frac{\| A \| + \| B \| }{\sqrt{T}} \right),
$$

and

$$
C_4 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{\hat{u}}_i \| , C_5 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{u}_i \| .
$$

Compactness of the set $\mathcal{X}$ implies existence of a finite constant $D_x$ that satisfies $\| x - y \| \leq D_x$ for all $x, y \in \mathcal{X}$. Since $x_t, \eta_t \in \mathcal{X}$ and by (21), we obtain

$$
\sum_{t=0}^{T} \| x_t - \bar{\theta}_t \| + L_x \sum_{t=0}^{T} \| x_{t+\mu} - \theta_{t+\mu} \| + L_y \sum_{t=0}^{T} \| u_t - \eta_t \|
$$

where

$$
C_0 = L_x \mu D_x , C_2 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{u}_i \| , C_3 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{\hat{u}}_i \| , C_4 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{\hat{u}}_i \| , C_5 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{u}_i \| .
$$

Inserting Lemma 8 (19), and (21) yields the desired result. The constants are given by $C_0 = L_x \mu D_x , C_2 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{u}_i \| , C_3 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{\hat{u}}_i \| , C_4 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{\hat{u}}_i \| , C_5 = \frac{1}{\sqrt{T}} \sum_{i=1}^{T} \| x^{u}_i \| .
$$

References

[1] M. Zinkevich, “Online convex programming and generalized infinitesimal gradient ascent,” in Proc. of the 20th International Conference on Machine Learning (ICML), pp. 928 – 936, 2003.

[2] S. Shalev-Shwartz, “Online learning and online convex optimization,” Foundations and Trends® in Machine Learning, vol. 4, no. 2, pp. 107–142, 2012.

[3] E. Hazan, “Introduction to online convex optimization,” Foundations and Trends® in Optimization, vol. 2, no. 3–4, pp. 157–225, 2016.

[4] A. Jadbabaie, A. Rakhlin, S. Shahrampour, and K. Sridharan, “Online Optimization : Competing with Dynamic Comparators,” in Proc. of the 18th International Conference on Artificial Intelligence and Statistics, vol. 38, pp. 398–406, 2015.

[5] O. Besbes, Y. Gur, and A. Zeevi, “Non-stationary stochastic optimization,” Operations Research, vol. 63, no. 5, pp. 1227–1244, 2015.

[6] A. Mohanty, S. Shahrampour, A. Jadbabaie, and A. Ribeiro, “Online optimization in dynamic environments: Improved regret rates for strongly convex problems,” in Proc. of the 55th Conference on Decision and Control (CDC), pp. 7195–7201, IEEE, 2016.

[7] M. Tanaka, “Real-time pricing with ramping costs: A new approach to managing a steep change in electricity demand,” Energy Policy, vol. 34, no. 18, pp. 3634–3643, 2006.

[8] Y. Li, G. Gu, and N. Li, “Using predictions in online optimization with switching costs: A fast algorithm and a fundamental limit,” in 2018 Annual American Control Conference (ACC), pp. 3008–3013, IEEE, 2018.

[9] Y. Li, X. Chen, and N. Li, “Online optimal control with linear dynamics and predictions: Algorithms and regret analysis,” in NeurIPS, pp. 14858–14870, 2019.

[10] M. Nomhoo and M. A. Müller, “Online gradient descent for linear dynamical systems,” in 21st IFAC World Congress, 2020. Available online at arXiv.org, arXiv:1912.09311.

[11] Y. Abbasi-Yadkori, P. Bartlett, and V. Kanade, “Tracking adversarial targets,” in Proc. of the 31st International Conference on Machine Learning, vol. 32, pp. 369–377, 2014.

[12] A. Cohen, A. Hasidim, T. Koren, N. Lazic, Y. Mansour, and K. Talwar, “Online linear quadratic control,” in Proc. of the 35th International Conference on Machine Learning, vol. 80, pp. 1029–1038, 2018.

[13] M. Akbari, B. Gharesifard, and T. Linder, “An Iterative Riccati Algorithm for Online Linear Quadratic Control,” arXiv e-prints, 2019. arXiv:1912.09451.

[14] N. Agarwal, B. Bullins, E. Hazan, S. Kakade, and K. Singh, “Online control with adversarial disturbances,” in Proc. of the 36th International Conference on Machine Learning, vol. 97, pp. 111–119, 2019.

[15] M. Colombino, E. Dall’Anese, and A. Bernstein, “Online optimization as a feedback controller: Stability and tracking,” IEEE Transactions on Control of Network Systems, vol. 7, no. 1, pp. 422–432, 2020.

[16] G. Bianchin, J. I. Poveda, and E. Dall’Anese, “Online optimization of switched LTI systems using continuous-time and hybrid accelerated gradient flows,” arXiv e-prints, 2020. arXiv:2008.03903.

[17] A. Hauswirth, S. Bolognani, G. Hug, and F. Dorfler, “Timescale separation in autonomous optimization,” IEEE Transactions on Automatic Control, vol. 66, no. 2, pp. 611–624, 2021.

[18] L. S. P. Lawrence, Z. E. Nelson, E. Mallada, and J. W. Simpson-Porco, “Optimal steady-state control for linear time-invariant systems,” in Proc. of the 2018 IEEE Conference on Decision and Control (CDC), pp. 3251–3257, 2018.

[19] Y. Nesterov, Lectures on Convex Optimization, vol. 137 of Springer Optimization and Its Applications. Springer International Publishing, 2 ed., 2018.

[20] T. Faulwasser, L. Grüne, and M. A. Müller, “Economic nonlinear model predictive control,” Foundations and Trends® in Systems and Control, vol. 5, no. 1, pp. 1–98, 2018.

[21] J.-P. Aubin, A. M. Bayen, and P. Saint-Pierre, Viability Theory, Springer-Verlag Berlin Heidelberg, 2 ed., 2011.

[22] A. Boccia, L. Grüne, and K. Worthmann, “Stability and feasibility of state constrained mpc without stabilizing terminal constraints,” Systems & Control Letters, vol. 72, pp. 14 – 21, 2014.

[23] J. B. Rawlings and D. Q. Mayne, Model Predictive Control: Theory and Design. Nob Hill Pub., 2009.
[24] J. N. Maidens, S. Kaynama, I. M. Mitchell, M. M. Oishi, and G. A. Dumont, “Lagrangian methods for approximating the viability kernel in high-dimensional systems,” *Automatica*, vol. 49, no. 7, pp. 2017 – 2029, 2013.

[25] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, “Multi-Parametric Toolbox 3.0,” in *Proc. of the 2013 European Control Conference (ECC)*, pp. 502–510, 2013.