

# A Computationally Governed Log-domain Interior-point Method for Model Predictive Control

Jordan Leung<sup>1</sup>, Frank Permenter<sup>2</sup>, and Ilya V. Kolmanovsky<sup>1</sup>

**Abstract**—This paper introduces a computationally efficient approach for solving Model Predictive Control (MPC) reference tracking problems with state and control constraints. The approach consists of three key components: First, a log-domain interior-point quadratic programming method that forms the basis of the overall approach; second, a method of warm-starting this optimizer by using the MPC solution from the previous timestep; and third, a computational governor that bounds the suboptimality of the warm-start by altering the reference command provided to the MPC problem. As a result, the closed-loop system is altered in a manner so that MPC solutions can be computed using fewer optimizer iterations per timestep. In a numerical experiment, the computational governor reduces the worst-case computation time of a standard MPC implementation by 90%, while maintaining good closed-loop performance.

## I. INTRODUCTION

Model Predictive Control (MPC) is a feedback strategy defined by the solution of a receding horizon Optimal Control Problem (OCP). MPC is widely used in both industrial and academic settings since it provides high-performance control while directly accounting for constraints. Additionally, a wide body of literature on the stability and robustness of MPC is available [1]–[4]. Unfortunately, implementing MPC can be challenging since a constrained OCP must be solved at every timestep. The development of efficient methods for solving these OCPs has helped address this challenge [5]–[8], but this still remains an open problem in applications with fast sampling rates and/or limited computing power.

A common approach to reduce the computational burden of MPC is to approximate the OCP solutions by performing a limited number of optimizer iterations per timestep - a procedure referred to as suboptimal MPC. While results pertaining to the stability and robustness of such methods do exist [9]–[11], computable certification bounds are limited to simple cases (e.g. input constrained linear systems [12]–[14]) or require significant modification of the OCP (e.g. constraint tightening [15]) to guarantee stability.

Anytime MPC methods are a class of suboptimal strategies that ensure stabilizing solutions can be computed under arbitrary time constraints. The approach in [16] ensures that a specified warm-starting scheme always decreases a Lyapunov

Toyota Research Institute (TRI) provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. The third author acknowledges support by the National Science Foundation award number CMMI-1904394.

<sup>1</sup> J. Leung and I. Kolmanovsky are with the University of Michigan, Ann Arbor, MI 48109, USA. Email: {jmlleung, ilya}@umich.edu. <sup>2</sup> F. Permenter is with the Toyota Research Institute. Email: frank.permenter@tri.global.

function through the addition of the so-called Lyapunov constraint to the OCP. Another approach uses relaxed barrier functions to ensure anytime stability at the cost of bounded constraint violation [17]. Alternatively, the approach in [18] guarantees convergence to a terminal set by solving convex feasibility problems.

This paper introduces an alternative approach for achieving computationally efficient MPC. The main novelty introduced in this paper is a computational governing scheme that alters the MPC reference command so that the suboptimality of the interior-point optimizer initialization is bounded. In other words, the proposed strategy modifies the MPC problem to ensure that nearly optimal solutions can be computed using fewer interior-point optimizer iterations per timestep.

The paper is organized as follows. Section II introduces the tracking MPC formulation used to generate control inputs. Section III provides a brief overview of the log-domain interior-point method (LDIPM) used for optimization, while Section IV describes how the LDIPM can be used to solve MPC problems. Section V introduces the computational governing strategy and Section VI demonstrates the efficacy of the computational governor through numerical experiments.

**Notation:** Let  $\mathbb{R}_{>0}^n$  denote the set of real  $n \times 1$  vectors, with strictly positive elements (define  $\mathbb{R}_{\geq 0}^n$  accordingly). Let  $\mathbb{Z}_+ = \mathbb{Z}_{\geq 0}$  represent the set of non-negative integers. Given  $a, b > 0$ , let  $\mathbb{N}_{[a,b]} = \mathbb{N} \cap [a, b]$ . Given  $x \in \mathbb{R}^n$  and  $W \in \mathbb{R}^{n \times n}$  with  $W \succ 0$ , let  $\|x\|_W = \sqrt{x^T W x}$  denote the  $W$ -norm. Let  $\|\cdot\|$  represent the 2-norm when no subscript is specified. Let  $\lambda_-(A)$  and  $\lambda_+(A)$  denote the minimum and maximum eigenvalues of  $A \in \mathbb{R}^{n \times n}$ . Given  $x \in \mathbb{R}^n$  and  $y \in \mathbb{R}^m$ , let  $(x \circ y) \in \mathbb{R}^n$  denote the elementwise multiplication, and  $e^x$  and  $x^{-1}$  denote the elementwise exponentiation and inverse. Let  $(x, y) = [x^T y^T]^T$ . Given  $x \in \mathbb{R}^n$ , let  $\text{diag}(x) = \text{diag}(x_1, \dots, x_n)$  denote the  $n \times n$  diagonal matrix containing  $x_i$  in the  $i^{\text{th}}$  diagonal element  $\forall i \in \mathbb{N}_{[1,n]}$ .

## II. PROBLEM SETTING

Consider the following Linear Time Invariant (LTI) system

$$x_{k+1} = Ax_k + Bu_k, \quad (1a)$$

$$y_k = Cx_k + Du_k, \quad (1b)$$

$$z_k = Ex_k + Fu_k, \quad (1c)$$

where  $k \in \mathbb{Z}_+$  is the discrete-time index,  $x_k \in \mathbb{R}^n$  is the state,  $u_k \in \mathbb{R}^{n_u}$  is the control,  $y_k \in \mathbb{R}^{n_y}$  is the constrained output, and  $z_k \in \mathbb{R}^{n_z}$  is the tracking output. The control objective is to drive the tracking output  $z_k$  to a desired

reference  $r \in \mathbb{R}^{n_z}$  subject to pointwise-in-time constraints

$$y_k \in \mathcal{Y}, \forall k \in \mathbb{Z}_+, \quad (2)$$

where  $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$  is a specified constraint set.

**Assumption 1.** *The pair  $(A, B)$  is stabilizable and the constraint set  $\mathcal{Y} = \{y \mid Yy \leq h\}$  is a compact polyhedron that contains the origin in its interior.*

Equilibria of (1) satisfy  $Z[x^T \ u^T \ z^T]^T = 0$ , where

$$Z = \begin{bmatrix} A - I & B & 0 \\ E & F & -I \end{bmatrix}, \quad (3)$$

and these equilibria can be parameterized by a reference command  $v \in \mathbb{R}^{n_v}$  according to  $(\bar{x}_v, \bar{u}_v, \bar{z}_v) = Gv$ , where  $G^T = [G_x^T \ G_u^T \ G_z^T]^T$  is a basis for the nullspace of  $Z$  and Assumption 1 ensures that  $\text{Null}(Z) \neq \{0\}$  [19], [20]. The following assumption excludes ill-posed reference tracking problems, e.g.,  $G_z = 0$ , and ensures that the reference uniquely determines the target equilibrium.

**Assumption 2.** *The matrix  $G_z$  is full rank and  $n_z = n_v$ .*

**Remark 1.** *The results herein could be extended to the case where  $G_z$  is full rank and  $n_z < n_v$  [19]. Assumption 2 is made instead to simplify the approach.*

The reference tracking MPC strategy of [20] is employed to solve the specified control problem. The following parametric OCP is used to generate the MPC feedback law:

$$\min_{(\xi, \mu)} \|\xi_N - \bar{x}_v\|_P^2 + \sum_{i=0}^{N-1} \|\xi_i - \bar{x}_v\|_Q^2 + \|\mu_i - \bar{u}_v\|_R^2 \quad (4a)$$

$$\text{s.t. } \xi_0 = x, \quad (4b)$$

$$\xi_{i+1} = A\xi_i + B\mu_i, \quad i \in \mathbb{N}_{[0, N-1]}, \quad (4c)$$

$$C\xi_i + D\mu_i \in \mathcal{Y}, \quad i \in \mathbb{N}_{[0, N-1]}, \quad (4d)$$

$$(\xi_N, v) \in \mathcal{T}, \quad (4e)$$

where  $N \in \mathbb{N}$  is the prediction horizon,  $\xi = (\xi_0, \dots, \xi_N)$  and  $\mu = (\mu_0, \dots, \mu_{N-1})$  are the predicted state and control sequences,  $P \in \mathbb{R}^{n_x \times n_x}$ ,  $Q \in \mathbb{R}^{n_x \times n_x}$ ,  $R \in \mathbb{R}^{n_u \times n_u}$  are weighting matrices, and  $\mathcal{T} \subseteq \mathbb{R}^{n+n_v}$  is a terminal set. The current state  $x$  and reference command  $v$  are parameters in this OCP. We use the notation  $\mathcal{P}_N(x, v)$  to refer to problem (4) specified with fixed parameters  $(x, v) \in \mathbb{R}^{n+n_v}$ . The following assumption ensures that (4) can be used to generate a stabilizing feedback law.

**Assumption 3.** *The cost matrices satisfy  $Q \succ 0$  and  $R \succ 0$ .*

**Remark 2.** *We assume  $Q \succ 0$ , as opposed to  $Q \succeq 0$ , in order to invoke a stability result [2, Theorem 13.1] for approximate MPC.*

Given  $Q$  and  $R$  specified according to Assumption 3, let  $P$  be positive-definite the solution to

$$P = Q + A^T P A - A^T P B K, \quad (5)$$

where  $K$  is the Linear Quadratic Regulator (LQR) gain

$$K = (R + B^T P B)^{-1} (B^T P A). \quad (6)$$

Further, let  $\mathcal{T} = \mathcal{O}_\infty$ , where  $\mathcal{O}_\infty \subseteq \mathbb{R}^{n+n_v}$  is the maximum constraint admissible set for the closed-loop system under LQR feedback [21]. That is,  $\mathcal{O}_\infty$  is the maximal set of pairs  $(x, v)$  such that if LQR is applied to (1) with  $x_0 = x$  and a constant reference  $v$ , then the constraint (2) is satisfied.

Since  $\mathcal{T} = \mathcal{O}_\infty$  is polyhedral under Assumption 1 [21], then  $\mathcal{P}_N(x, v)$  can be written in a condensed form:

$$\min_{\mu} \frac{1}{2} \mu^T H \mu + \mu^T W \theta \quad (7a)$$

$$\text{s.t. } M\mu + L\theta + b \geq 0, \quad (7b)$$

where  $\theta = (x, v)$  and  $H \succ 0, W, M, L, b$  are defined in [19] as a function of the problem data in (4). The set of feasible parameters for (7) is

$$\Gamma_N = \{\theta \in \mathbb{R}^{n+n_v} \mid \exists \mu : M\mu + L\theta + b \geq 0\}, \quad (8)$$

which is equivalent to the  $N$ -step backwards reachable set to  $\mathcal{O}_\infty$ . Note that both  $\mathcal{O}_\infty$  and  $\Gamma_N$  are polyhedral sets with representations that can be computed offline [19], [21]. The following set-valued maps are defined for convenience

$$\mathcal{O}_\infty(v) = \{x \in \mathbb{R}^n \mid (x, v) \in \mathcal{O}_\infty\}, \quad (9)$$

$$\Gamma_N(v) = \{x \in \mathbb{R}^n \mid (x, v) \in \Gamma_N\}. \quad (10)$$

The MPC feedback policy is defined by

$$u^*(x, v) = \Xi \mu^*(x, v), \quad (11)$$

where  $\mu^*(x, v)$  is the optimal solution to  $\mathcal{P}_N(x, v)$  and  $\Xi = [I_{n_u} \ 0 \ \dots \ 0]$  is a matrix that selects the first control input. The following theorem details the stability and convergence properties of the closed-loop MPC system for a constant reference  $v$ .

**Theorem 1.** *([4, Theorem 4.4.2], [19, Theorem 1]) Let Assumptions 1-3 hold and consider the closed-loop system*

$$x_{k+1} = Ax_k + Bu^*(x_k, v), \quad (12)$$

*starting from an initial condition  $x_0$  with a constant reference command  $v$ . Then for all  $(x_0, v) \in \Gamma_N$ , all solutions satisfy:  $(x_k, v) \in \Gamma_N, \forall k \in \mathbb{Z}_+$ ;  $y_k \in \mathcal{Y}, \forall k \in \mathbb{Z}_+$ ; and  $\lim_{k \rightarrow \infty} x_k = \bar{x}_v$ . If, in addition,  $v \in \text{Int } \mathcal{V}$  then  $\bar{x}_v$  is asymptotically stable, where  $\mathcal{V} = \{v \mid G_y v \in \mathcal{Y}\}$  is the set of constraint admissible references and  $G_y = CG_x + DG_u$ .*

**Assumption 4.** *The target reference satisfies  $r \in \text{Int } \mathcal{V}$ .*

Clearly, the feedback policy (11) is sufficient to solve the given control problem. However, computing the solution of (7) at each timestep may be difficult in applications with fast sampling rates and/or limited computing power. To this end, a computationally governed optimization approach for solving (7) is proposed in this paper.

### III. A LOG-DOMAIN INTERIOR-POINT METHOD FOR QUADRATIC PROGRAMMING

In this section, a brief overview of the log-domain interior-point method (LDIPM) from [22] is presented. To avoid

dealing with the extra parameters in (7), we consider a generic convex quadratic program (QP) of the form:

$$\min_z \frac{1}{2} z^T H z + c^T z \quad (13a)$$

$$\text{s.t.} \quad A z + b \geq 0, \quad (13b)$$

where  $z \in \mathbb{R}^p$  is the vector of decision variables, and  $A \in \mathbb{R}^{m \times p}$ ,  $b \in \mathbb{R}^m$ ,  $H \in \mathbb{R}^{p \times p}$ , and  $c \in \mathbb{R}^p$  are the problem data. It is assumed that  $H \succeq 0$ , there exists  $z \in \mathbb{R}^p$  satisfying  $A z + b > 0$  for all  $\beta \in \mathbb{R}$ , the sublevel set  $\{z \mid \frac{1}{2} z^T H z + c^T z \leq \beta, A z + b \geq 0\}$  is bounded, and  $A^T A + H \succ 0$ .

Consider the following *central-path* equations for (13)

$$A^T \lambda = H z + c, \quad s = A z + b, \quad (14a)$$

$$\lambda \geq 0, \quad s \geq 0, \quad s_i \lambda_i = \eta, \quad \forall i \in \mathbb{N}_{[1, m]}, \quad (14b)$$

where  $\eta > 0$  is a fixed homotopy parameter,  $s \in \mathbb{R}^m$  is the constraint slack, and  $\lambda \in \mathbb{R}^m$  is the vector of dual variables. Note that when  $\eta = 0$ , these equations reduce to the Karush-Kuhn-Tucker (KKT) optimality conditions for (13). Next, consider the following logarithmic change-of-variables. Let  $\gamma \in \mathbb{R}^m$  and define  $\lambda = \sqrt{\eta} e^\gamma$  and  $s = \sqrt{\eta} e^{-\gamma}$ , such that the *log-domain central-path* equations are

$$\sqrt{\eta} A^T e^\gamma = H z + c, \quad \sqrt{\eta} e^{-\gamma} = A z + b. \quad (15)$$

Note that the conditions in (14b) are automatically satisfied by the change-of-variables. We define the *central-path* as the map  $\eta \mapsto (z, \gamma)$  such that  $(z, \gamma, \eta)$  satisfy (15).

The LDIPM solves (13) by applying Newton's method to (15) with a decreasing sequence of  $\eta$ . The Newton direction  $d = d(\gamma, \eta) \in \mathbb{R}^m$  of the LDIPM satisfies

$$\sqrt{\eta} A^T (e^\gamma + e^\gamma \circ d) = H z + c, \quad (16a)$$

$$\sqrt{\eta} (e^{-\gamma} - e^{-\gamma} \circ d) = A z + b. \quad (16b)$$

The following theorem establishes uniqueness of  $d(\gamma, \eta)$  and  $z(\gamma, \eta)$ , and provides a method of computing both.

**Theorem 2.** ([22, Theorem 2.1]) *For all  $\gamma \in \mathbb{R}^m$  and  $\eta > 0$ , the Newton direction  $d = d(\gamma, \eta)$  and the decision variable  $z = z(\gamma, \eta)$  satisfy*

$$d = \mathbf{1} - \frac{1}{\sqrt{\eta}} e^\gamma \circ (A z + b), \quad (17)$$

$$(A^T \Phi(\gamma) A + H) z = 2\sqrt{\eta} A^T e^\gamma - (c + A^T \Phi(\gamma) b), \quad (18)$$

where  $\mathbf{1} \in \mathbb{R}^m$  is a vector of ones and  $\Phi(\gamma) = \text{diag}(e^{2\gamma})$ . Moreover,  $A^T \Phi(\gamma) A + H \succ 0$ .

Given  $\eta > 0$ , iterates generated by the update rule

$$\gamma_{i+1} = \gamma_i + \frac{1}{\alpha_i} d(\gamma_i, \eta), \quad (19)$$

$$\alpha_i = \max \{1, \|d(\gamma_i, \eta)\|_\infty^2\}, \quad (20)$$

are globally convergent to the central-path point  $(z, \gamma, \eta)$  [22]. The following lemma describes conditions under which iterates are primal-dual feasible with bounded suboptimality.

**Lemma 1.** ([22, Lemma 3.3]) *For  $\eta > 0$ , let  $d = d(\gamma, \eta)$  and  $z = z(\gamma, \eta)$ . Let  $\lambda = \sqrt{\eta}(e^\gamma + e^\gamma \circ d)$  and  $s = \sqrt{\eta}(e^{-\gamma} -$*

*$e^{-\gamma} \circ d)$ . If  $\|d\|_\infty \leq 1$ , then  $(z, s, \lambda)$  satisfy the primal-dual feasibility conditions*

$$A z + b = s, \quad A^T \lambda = H z + c, \quad \lambda \geq 0, \quad s \geq 0.$$

*Further,  $\|s \circ \lambda\|_1 = \eta(m - \|d\|^2)$ .*

---

**Algorithm 1** Longstep( $H, c, A, b, \gamma_0, \eta_0, \eta_f$ )

---

- 1:  $\gamma \leftarrow \gamma_0, \eta \leftarrow \eta_0$
  - 2: **while**  $\eta > \eta_f$  **or**  $\|d(\gamma, \eta)\|_\infty > 1$  **do**
  - 3:    $\eta \leftarrow \min\{\eta, \inf\{\eta > 0 : \|d(\gamma, \eta)\|_\infty \leq 1\}\}$
  - 4:    $\alpha \leftarrow \max\{1, \|d(\gamma, \eta)\|_\infty^2\}$
  - 5:    $\gamma \leftarrow \gamma + \frac{1}{\alpha} d(\gamma, \eta)$
  - 6: **end while**
  - 7:  $z \leftarrow (A^T \Phi(\gamma) A + H)^{-1} [2\sqrt{\eta} A^T e^\gamma - (c + A^T \Phi(\gamma) b)]$ ,
  - 8: **return**  $(z, \gamma, \eta)$
- 

Lemma 1 motivates the longstep procedure described in Algorithm 1. At each iteration, the longstep procedure seeks to reduce  $\eta$  by computing

$$\eta^* = \inf\{\eta > 0 \mid \|d(\gamma, \eta)\|_\infty \leq 1\}, \quad (21)$$

where we define  $\eta^* = \infty$  when the set in (21) is empty. When  $\eta^* < \infty$  is found, Algorithm 1 provides an update that ensures  $(z, \gamma, \eta)$  is primal-dual feasible and within a neighborhood of the central-path. Note that  $\eta^*$  can be computed in  $O(m)$  time by iterating through  $2m$  linear inequalities [22, Section 3.2.1]. Moreover, the following theorem shows that Algorithm 1 terminates globally.

**Theorem 3.** ([22, Theorem 3.2]) *For any input  $(\gamma_0, \eta_0, \eta_f) \in \mathbb{R}^m \times \mathbb{R}_{>0} \times \mathbb{R}_{>0}$ , Algorithm 1 terminates and returns  $(z, \gamma, \eta)$  with  $\eta \leq \eta_f$ ,  $\|d(\gamma, \eta)\|_\infty \leq 1$ , and*

$$A z + b \geq 0, \quad \frac{1}{2} z^T W z + c^T z \leq V^* + m \eta_f,$$

where  $V^*$  denotes the optimal value of the QP (13).

#### IV. APPLICATION OF THE LOG-DOMAIN INTERIOR POINT METHOD TO MPC

We now consider the application of LDIPM to the MPC problem in Section II. The following theorem and corollary demonstrate that the equilibrium  $\bar{x}_v$  is asymptotically stable when MPC solutions are computed using Algorithm 1 with a sufficiently small truncation tolerance  $\eta_f$ .

**Theorem 4.** *Let Assumptions 1-3 hold. Let  $\tilde{\mu} : (x, v) \mapsto \mu$  be a function that generates a feasible solution to  $\mathcal{P}_N(x, v)$  satisfying*

$$J(x, v, \tilde{\mu}(x, v)) < V(x, v) + \|x - \bar{x}_v\|_Q^2, \quad (22)$$

for all  $(x, v) \in \Gamma_N \setminus (\bar{x}_v, v)$ , where  $J$  and  $V$  are the cost function and optimal value function of  $\mathcal{P}_N(x, v)$ . Then, consider the closed-loop dynamics

$$x_{k+1} = A x_k + B \Xi \tilde{\mu}(x_k, v), \quad (23)$$

starting from an initial condition  $x_0$  with a constant reference command  $v$ . Then for all  $(x_0, v) \in \Gamma_N$ , all solutions satisfy:

$(x_k, v) \in \Gamma_N, \forall k \in \mathbb{Z}_+; y_k \in \mathcal{Y}, \forall k \in \mathbb{Z}_+; \text{ and } \lim_{k \rightarrow \infty} x_k = \bar{x}_v. \text{ If, in addition, } v \in \text{Int } \mathcal{V} \text{ then } \bar{x}_v \text{ is asymptotically stable.}$

*Proof.* The result is a direct consequence of Theorem 1 and [2, Theorem 13.1].  $\square$

**Corollary 1.** *Suppose Algorithm 1 is applied to  $\mathcal{P}_N(x_k, v)$  and let  $\eta_{f,k}$  represent the truncation tolerance  $\eta_f$  specified in Algorithm 1 at timestep  $k$ . Further, let  $\eta_{f,k}$  satisfy*

$$m\eta_{f,k} < \|x_k - \bar{x}_v\|_Q^2, \quad \forall k \geq 0, \quad (24)$$

where  $m \in \mathbb{N}$  is the number of constraints in (7) (i.e.  $b \in \mathbb{R}^m$ ). Then, any output of Algorithm 1 satisfies Theorem 4.

*Proof.* Any solution  $\mu$  generated by Algorithm 1 is feasible and satisfies  $J(x, v, \mu) - V(x, v) \leq m\eta_{f,k}$  by Theorem 3. Hence, (22) is satisfied by the bound in (24).  $\square$

**Assumption 5.** *At all timesteps  $k$ , the LDIPM truncation tolerance  $\eta_{f,k}$  is chosen to satisfy (24).*

Now we consider how the LDIPM can be warm-started. Let  $\xi_{k-1} = (\xi_{0,k-1}, \dots, \xi_{N,k-1})$  and  $\mu_{k-1} = (\mu_{0,k-1}, \dots, \mu_{N-1,k-1})$  represent the state and control sequences outputted by the LDIPM applied to  $\mathcal{P}_N(x_{k-1}, v)$ . Denote by  $\eta_{k-1} \leq \eta_{f,k-1}$  the tolerance that Algorithm 1 truncated with at timestep  $k-1$ . The warm-started primal decision variable at timestep  $k$  is generated by

$$\bar{\mu}_k = (\mu_{1,k-1}, \dots, \mu_{N-1,k-1}, \bar{u}_v - K(\xi_{N,k-1} - \bar{x}_v)), \quad (25)$$

where  $K$  is the LQR gain in (6). Note that  $\bar{\mu}_k$  is always a feasible solution candidate to  $\mathcal{P}_N(x_k, v)$  as consequence of the MPC formulation [1, Chapter 2] and feasibility of  $\mu_{k-1}$  at  $k-1$ . Thus, the corresponding slack variable satisfies  $\bar{s}_k = M\bar{\mu}_k + L\theta_k + b \geq 0$ , where  $\theta_k = (x_k, v)$ . The log-domain variable used to initialize Algorithm 1 is then generated by

$$\bar{\gamma}_k = -\log \left( \max \left\{ \frac{\bar{s}_k}{\sqrt{\eta_{k-1}}}, \epsilon_s \mathbf{1} \right\} \right), \quad (26)$$

where  $\epsilon_s > 0$  is a small tolerance, log operates elementwise, and  $\max\{x, y\} = (\max(x_1, y_1), \dots, \max(x_n, y_n)) \in \mathbb{R}^n$  for  $x, y \in \mathbb{R}^n$ . The first argument in the max operator is a rearrangement of the parameterization in (15), whereas the second argument is included to ensure that (26) is defined when  $\bar{s}_k$  has elements that are equal to zero.

Once initialized with  $\bar{\gamma}_k$ , Algorithm 1 begins by computing  $\eta^*$  in (21). That is, it finds the smallest  $\eta$  that ensures the warm-start is within a neighborhood of the corresponding central-path point. In this sense, the warm-started solution only needs to be sufficiently close to *some* location on the central-path. In contrast, other IPMs that are not designed with warm-starting in mind may be forced to start with a fixed (often large) initial value of  $\eta$ .

Note that the warm-start in (26) may be a poor initial guess if a large reference change occurs. In such cases, the LDIPM may require several iterations to converge. The following section introduces an approach for altering the reference command to avoid this problem.

## V. A COMPUTATIONAL GOVERNOR FOR THE LOG-DOMAIN INTERIOR-POINT METHOD

Next, consider an MPC feedback law defined by solving  $\mathcal{P}_N(x_k, v_k)$  with a changing reference command defined by

$$v_k = v_{k-1} + \kappa_k(r - v_{k-1}), \quad (27)$$

where  $r$  is the desired reference and  $\kappa_k \in [0, 1]$  is a time-varying parameter that dictates the rate at which  $v_k$  converges to  $r$ . The parameterization in (27) is commonly used in Scalar Reference Governors (SRGs), where  $\kappa_k$  is maximized at each timestep subject to the constraint that  $(x_k, v_k)$  stays inside of an invariant constraint admissible set [23].

This section introduces a *computational governor* (CG) that chooses  $\kappa_k$  at each timestep subject to restrictions on the suboptimality and feasibility of the warm-start. In other words, the CG enforces a computational constraint in a similar approach to how SRGs enforce system constraints. To facilitate this development, note that the LDIPM Newton step can be parameterized by  $\eta$  and  $\kappa$  in the following manner.

**Proposition 1.** *Let the reference command  $v = v_k$  in (7) be defined by (27) and let  $d(\gamma, \eta, \kappa)$  represent the LDIPM Newton step for (7) at a given  $\gamma \in \mathbb{R}^m, \eta > 0$ , and  $\kappa \in [0, 1]$ . Then, there exist  $d_0(\gamma), d_1(\gamma), d_2(\gamma) \in \mathbb{R}^m$  such that*

$$d(\gamma, \eta, \kappa) = d_0(\gamma) + d_1(\gamma) \frac{1}{\sqrt{\eta}} + d_2(\gamma) \frac{1}{\sqrt{\eta}} \kappa. \quad (28)$$

*Proof.* Note that when (17) and (18) in Theorem 2 are written for the condensed OCP in (7), one obtains

$$d = \mathbf{1} - \frac{1}{\sqrt{\eta}} e^\gamma \circ (M\mu + L_x x + L_v v + b), \quad (29)$$

$$(M^T \Phi(\gamma) M + H) \frac{1}{\sqrt{\eta}} \mu = 2M^T e^\gamma - \frac{1}{\sqrt{\eta}} (W_x x + W_v v) - \frac{1}{\sqrt{\eta}} M^T \Phi(\gamma) (L_x x + L_v v + b), \quad (30)$$

where  $L\theta = L_x x + L_v v$  and  $W\theta = W_x x + W_v v$ . Then by considering the parameterization of  $v$  in (27) and observing that  $\frac{1}{\sqrt{\eta}} \mu$  is affine with respect to  $(\frac{1}{\sqrt{\eta}}, \frac{1}{\sqrt{\eta}} \kappa)$  and  $d$  is affine with respect to  $(\frac{1}{\sqrt{\eta}} \mu, \frac{1}{\sqrt{\eta}}, \frac{1}{\sqrt{\eta}} \kappa)$ , it follows that  $d$  is affine with respect to  $(\frac{1}{\sqrt{\eta}}, \frac{1}{\sqrt{\eta}} \kappa)$ .  $\square$

**Remark 3.** *An efficient procedure for computing  $d_0, d_1, d_2$  is provided in the Appendix.*

As a consequence of Proposition 1, the solution of the following optimization problem can be used to specify  $(\eta, \kappa)$  prior to the start of Algorithm 1 given a warm-start  $\bar{\gamma}_k$

$$\max_{\eta, \kappa} \quad \kappa - c\sqrt{\eta} \quad (31a)$$

$$\text{s.t.} \quad \|d(\bar{\gamma}_k, \eta, \kappa)\|_\infty \leq 1, \quad (31b)$$

$$\eta \in [\eta_{\min}, \eta_{\max}], \quad (31c)$$

$$\kappa \in [0, 1], \quad (31d)$$

where  $c \geq 0$  is a weighting parameter. When  $c = 0$ , solving (31) finds the largest reference step  $\kappa^*$  that satisfies the primal-dual feasibility conditions of Lemma 1. Setting  $c > 0$

reduces the reference step but improves optimality of the warm-start.

Note that (31) can be expressed as a two-dimensional linear program (LP) in the variables  $\sqrt{\eta}$  and  $\kappa$  since (31b) is equivalent to

$$(d_0 - 1)\sqrt{\mu} + d_2\kappa \leq -d_1, \quad -(d_0 + 1)\sqrt{\mu} - d_2\kappa \leq d_1.$$

To solve the resulting LP, we use Seidel's algorithm described in [24]. We choose this algorithm because of its capability of solving low-dimensional LPs with  $m$  constraints in  $O(m)$  time.

In summary, we propose the following procedure for efficiently computing MPC control actions at each timestep

- 1) *Warm-start*: Initialize  $\bar{\gamma}_k$  according to (26),
- 2) *Computational governor*: Set  $(\eta_k, \kappa_k)$  to the solution of (31) when feasible, otherwise set  $(\eta_k, \kappa_k) = (\bar{\eta}, 0)$  where  $\bar{\eta} \gg 1$  is a large constant value,
- 3) *LDIPM*: Solve  $\mathcal{P}_N(x_k, v_k)$  using Algorithm 1 starting with a penalty parameter  $\eta_k$  and log-space variable  $\bar{\gamma}_k$ .

## VI. NUMERICAL EXAMPLES

The linear bicycle model in [25, Section II-A] is used to demonstrate the efficacy of the computationally governed LDIPM. The system states are  $x = (\beta, r, y)$  where  $\beta$  is the ratio of lateral to longitudinal velocity,  $r$  is the yaw rate, and  $y$  is the lateral position. The control input is the steering angle  $u = \delta$ . The continuous-time state-space matrices  $(A_c, B_c)$  are

$$\left( \begin{bmatrix} \frac{-(C_{af} + C_{ar})}{mU_x} & \frac{-(aC_{af} - bC_{ar})}{mU_x^2} - 1 & 0 \\ \frac{-(aC_{af} - bC_{ar})}{I_{zz}} & \frac{-(a^2C_{af} + b^2C_{ar})}{I_{zz}U_x} & 0 \\ U_x & 0 & 0 \end{bmatrix}, \begin{bmatrix} C_{af} \\ mU_x \\ aC_{af} \\ I_{zz} \\ 0 \end{bmatrix} \right)$$

where  $U_x = 10$  is the constant longitudinal velocity,  $m$  is the vehicle mass,  $I_{zz}$  is the yaw moment of inertia,  $C_{af}$  and  $C_{ar}$  are the front and rear cornering stiffness parameters, and  $a$  and  $b$  are the front and rear axle-CG distances. The values used for these parameters are those in [25, Table I]. The system is controlled using the MPC feedback law generated by solving  $\mathcal{P}_N(x_k, v_k)$  using the three-step strategy described in Section V. The MPC law is defined using a sampling period of  $T = 0.1$ , a horizon of  $N = 10$ , and weight matrices  $Q = \text{diag}(1, 1, 10)$  and  $R = 1$ . The CG in (31) is executed with parameters  $c = 1$ ,  $\eta_{\min} = 10^{-10}$ , and  $\eta_{\max} = 10^{-2}$ . The system constraints are  $\beta \in [-0.2, 0.2]$ ,  $r \in [-4, 4]$ ,  $y \in [-4, 4]$ , and  $\delta \in [-1, 1]$ .

Figure 1 compares the system response with and without the CG. When no CG is used, several iterations are needed to converge to a solution when the setpoint changes at  $t = 0$  and  $t = 10$ . Meanwhile, MPC solutions are obtained using only a single iteration per timestep when the CG is added to the closed-loop system. This is due to the CG maintaining a low value of  $\eta^*$ , thus Algorithm 1 is initialized near the optimal solution at each timestep. Moreover, the CG introduces very little degradation in the settling time of the controller and the solution operates near the constraint boundary of  $\delta$ .

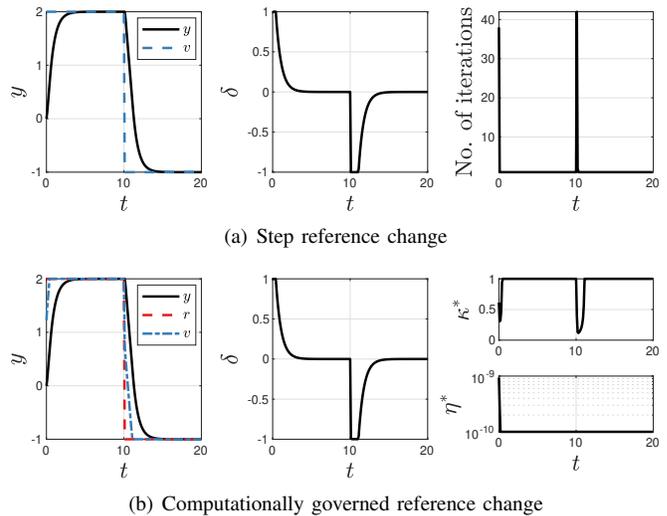


Fig. 1. Selecting  $(\eta^*, \kappa^*)$  using the computational governor allows for MPC solutions to be computed using only 1 iteration per timestep.

To quantify the reduction in worst-case computation time when the CG is used, the experiments in Figure 1 were repeated 1000 times using a 2018 MacBook Pro running MATLAB R2019b. The worst-case computation times for Algorithm 1 in Case (a) and the combination of Algorithms 1 and solving (31) in Case (b) were recorded for each trial. Over the 1000 trials, the average of these times were (a)  $10.8 \pm 0.9$  ms and (b)  $1.0 \pm 0.1$  ms. Thus, the worst-case computation time is reduced by approximately 90% when the CG is used.

## VII. CONCLUSIONS

This paper introduced a computationally efficient method of implementing referencing tracking MPC. The main novelty introduced in this paper is a computational governing scheme that alters the MPC reference command so that the suboptimality of the optimizer initialization is bounded. Examples demonstrated that using this strategy can reduce the worst-case computation time of MPC by 90%. Future work will be devoted to deriving theoretical guarantees for this strategy such as finite-time convergence of the reference command and asymptotic stability of the desired equilibrium.

### APPENDIX: COMPUTATION OF $(d_0, d_1, d_2)$

Let  $\gamma \in \mathbb{R}^m$  be a fixed log-domain variable and define constants  $\eta_1, \eta_2 > 0$ ,  $\kappa_1, \kappa_2 \in [0, 1]$ ,  $\eta_1 \neq \eta_2$ ,  $\kappa_1 \neq \kappa_2$ . Consider the Newton direction equation (16a) for (7) given each permutation of these parameters and a reference defined by  $v' = v + \kappa(r - v)$ , i.e.,

$$\begin{aligned} \sqrt{\eta_1} A^T e^\gamma \circ (\mathbf{1} + \hat{d}_1) &= H\mu_1 + W_x x + W_v [v + \kappa_1(r - v)] \\ \sqrt{\eta_1} A^T e^\gamma \circ (\mathbf{1} + \hat{d}_2) &= H\mu_2 + W_x x + W_v [v + \kappa_2(r - v)] \\ \sqrt{\eta_2} A^T e^\gamma \circ (\mathbf{1} + \hat{d}_3) &= H\mu_3 + W_x x + W_v [v + \kappa_1(r - v)] \\ \sqrt{\eta_2} A^T e^\gamma \circ (\mathbf{1} + \hat{d}_4) &= H\mu_4 + W_x x + W_v [v + \kappa_2(r - v)] \end{aligned}$$

By combining these four equations and using some algebraic manipulation, one can arrive at the equation,

$$\sqrt{\eta}A^T e^\gamma \circ (\mathbf{1} + d) = H\mu + W_x x + W_v[v + \kappa(r - v)], \quad (32)$$

where

$$\begin{aligned} \kappa &= p\kappa_1 + (1-p)\kappa_2, & \mu &= q\bar{\mu}_1 + (1-q)\bar{\mu}_2, \\ d &= q\bar{d}_1 + (1-q)\bar{d}_2, & \frac{1}{\sqrt{\eta}} &= q\frac{1}{\sqrt{\eta_1}} + (1-q)\frac{1}{\sqrt{\eta_2}}, \\ \bar{d}_1 &= p\hat{d}_1 + (1-p)\hat{d}_2, & \bar{\mu}_1 &= p\mu_1 + (1-p)\mu_2, \\ \bar{d}_2 &= p\hat{d}_3 + (1-p)\hat{d}_4, & \bar{\mu}_2 &= p\mu_3 + (1-p)\mu_4, \end{aligned} \quad (33)$$

and where  $p, q \in \mathbb{R}$  are constants that we will specify later. Repeating the exact same procedure, but instead starting with the four equations for (16b) gives

$$\sqrt{\eta}e^{-\gamma} \circ (\mathbf{1} - d) = M\mu + b + L_x x + L_v[v_0 + \kappa(r - v_0)]. \quad (34)$$

By comparing (32) and (34) to (16a) and (16b), one can observe that  $\mu$  and  $d$  defined in (33) are the primal variable and Newton step for parameters  $\kappa$  and  $\eta$  defined in (33).

So, for a given  $(\eta, \kappa)$  one can solve for the Newton step  $d(\gamma, \eta, \kappa)$  by defining  $p = a_0 + a_1\kappa$  and  $q = b_0 + b_1\eta^{-1/2}$ , where  $a_0 = -\kappa_2(\kappa_1 - \kappa_2)^{-1}$ ,  $a_1 = (\kappa_1 - \kappa_2)^{-1}$ ,  $b_0 = -\eta_2^{-1/2}(\eta_1^{-1/2} - \eta_2^{-1/2})^{-1}$ ,  $b_1 = (\eta_1^{-1/2} - \eta_2^{-1/2})^{-1}$ . Substituting these expressions for  $p$  and  $q$  into the equation for  $d$  in (33) yields, after some algebraic manipulation is performed,

$$d = d_0 + d_1 \frac{1}{\sqrt{\eta}} + d_2 \frac{\kappa}{\sqrt{\eta}} + d_3 \kappa, \quad (35)$$

where

$$\begin{aligned} d_0 &= b_0 c_1 + (1 - b_0) c_3, & d_1 &= b_1 (c_1 - c_3), \\ d_2 &= b_1 (c_2 - c_4), & d_3 &= b_0 c_2 + (1 - b_0) c_4, \\ c_1 &= a_0 \hat{d}_1 + (1 - a_0) \hat{d}_2, & c_2 &= a_1 (\hat{d}_1 - \hat{d}_2), \\ c_3 &= a_0 \hat{d}_3 + (1 - a_0) \hat{d}_4, & c_4 &= a_1 (\hat{d}_3 - \hat{d}_4). \end{aligned} \quad (36)$$

Further, we must have that  $d_3 = 0$  according to (28), so we can eliminate the need to directly compute one Newton direction (e.g.  $\hat{d}_4$ ) by using the equality  $d_3 = 0$  to obtain

$$\hat{d}_4 = b_0(1 - b_0)^{-1}(\hat{d}_1 - \hat{d}_2) + \hat{d}_3. \quad (37)$$

Thus,  $d_0$ ,  $d_1$ , and  $d_2$  in Proposition 1 can be computed by defining constants  $\eta_1, \eta_2, \kappa_1, \kappa_2$  and computing the sampled Newton direction  $\hat{d}_i$  for three of the four permutations by exploiting the common factorization.

## REFERENCES

- [1] J. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Madison, WI: Nob Hill Publishing, 2009.
- [2] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.
- [3] B. Kouvaritakis and M. Cannon, "Model predictive control: Classical, robust and stochastic," *Switzerland: Springer International Publishing*, 2016.
- [4] G. C. Goodwin, M. M. Seron, and J. A. De Doná, *Constrained Control and Estimation: An Optimisation Approach*. Springer Publishing Company, Incorporated, 1st ed., 2010.
- [5] A. Domahidi, E. Chu, and S. P. Boyd, "ECOS: An SOCP solver for embedded systems," *2013 European Control Conference (ECC)*, pp. 3071–3076, 2013.
- [6] P. Patrinos and A. Bemporad, "An accelerated dual gradient-projection algorithm for embedded linear model predictive control," *IEEE Transactions on Automatic Control*, vol. 59, no. 1, pp. 18–33, 2014.
- [7] D. Liao-McPherson and I. Kolmanovsky, "FBstab: A proximally stabilized semismooth algorithm for convex quadratic programming," *Automatica*, vol. 113, p. 108801, 2020.
- [8] G. Frison and M. Diehl, "HPIPM: a high-performance quadratic programming framework for model predictive control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6563–6569, 2020. 21st IFAC World Congress.
- [9] M. Diehl, R. Findeisen, F. Allgower, H. G. Bock, and J. P. Schlöder, "Nominal stability of real-time iteration scheme for nonlinear model predictive control," *IEE Proceedings - Control Theory and Applications*, vol. 152, pp. 296–308, May 2005.
- [10] D. Liao-McPherson, M. Nicotra, and I. Kolmanovsky, "Time-distributed optimization for real-time model predictive control: Stability, robustness, and constraint satisfaction," *Automatica*, vol. 117, p. 108973, 2020.
- [11] A. Zanelli, Q. Tran-Dinh, and M. Diehl, "A Lyapunov function for the combined system-optimizer dynamics in inexact model predictive control," *Automatica*, vol. 134, p. 109901, 2021.
- [12] D. Liao-McPherson, T. Skibik, J. Leung, I. V. Kolmanovsky, and M. M. Nicotra, "An analysis of closed-loop stability for linear model predictive control based on time-distributed optimization," *IEEE Transactions on Automatic Control*, pp. 1–1, 2021.
- [13] J. Leung, D. Liao-McPherson, and I. V. Kolmanovsky, "A computable plant-optimizer region of attraction estimate for time-distributed linear model predictive control," in *2021 American Control Conference (ACC)*, pp. 3384–3391, 2021.
- [14] S. Richter, C. N. Jones, and M. Morari, "Computational complexity certification for real-time MPC with input constraints based on the fast gradient method," *IEEE Transactions on Automatic Control*, vol. 57, no. 6, pp. 1391–1403, 2011.
- [15] M. Rubagotti, P. Patrinos, and A. Bemporad, "Stabilizing linear model predictive control under inexact numerical optimization," *IEEE Transactions on Automatic Control*, vol. 59, pp. 1660–1666, June 2014.
- [16] M. N. Zeilinger, D. M. Raimondo, A. Domahidi, M. Morari, and C. N. Jones, "On real-time robust model predictive control," *Automatica*, vol. 50, no. 3, pp. 683–694, 2014.
- [17] C. Feller and C. Ebenbauer, "A stabilizing iteration scheme for model predictive control based on relaxed barrier functions," *Automatica*, vol. 80, pp. 328–339, 2017.
- [18] A. Bemporad, D. Bernardini, and P. Patrinos, "A convex feasibility approach to anytime model predictive control," *arXiv preprint arXiv:1502.07974*, 2015.
- [19] T. Skibik, D. Liao-Mc Pherson, T. Cunis, I. V. Kolmanovsky, and M. M. Nicotra, "A feasibility governor for enlarging the region of attraction of linear model predictive controllers," *IEEE Transactions on Automatic Control*, 2021.
- [20] D. Limon, I. Alvarado, T. Alamo, and E. Camacho, "MPC for tracking piecewise constant references for constrained linear systems," *Automatica*, vol. 44, no. 9, pp. 2382–2387, 2008.
- [21] E. G. Gilbert and K. T. Tan, "Linear systems with state and control constraints: the theory and application of maximal output admissible sets," *IEEE Transactions on Automatic Control*, vol. 36, no. 9, pp. 1008–1020, 1991.
- [22] F. Permenter, "Log-domain interior-point methods for quadratic programming," 2021. Online at [http://www.optimization-online.org/DB\\_HTML/2021/09/8600.html](http://www.optimization-online.org/DB_HTML/2021/09/8600.html).
- [23] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, 2017.
- [24] R. Seidel, "Small-dimensional linear programming and convex hulls made easy," *Discrete & Computational Geometry*, vol. 6, no. 3, pp. 423–434, 1991.
- [25] C. E. Beal and J. C. Gerdes, "Model predictive control for vehicle stabilization at the limits of handling," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1258–1269, 2013.