2025 23rd European Control Conference (ECC)
June 24-27, 2025. Thessaloniki, Greece

Split-as-a-Pro: behavioral control via operator splitting and alternating projections

Yu Tang, Carlo Cenedese, Alessio Rimoldi, Florian Dörfler, John Lygeros, Alberto Padoan

***Abstract*— The paper introduces Split-as-a-Pro, a control framework that integrates behavioral systems theory, operator splitting methods, and alternating projection algorithms. The framework reduces dynamic optimization problems — arising in both control and estimation — to efficient projection computations. Split-as-a-Pro builds on a non-parametric formulation that exploits system structure to separate dynamic constraints imposed by individual subsystems from external ones — such as interconnection constraints and input/output constraints. This enables the use of arbitrary system representations, as long as the associated projection is efficiently computable, thereby enhancing scalability and compatibility with gray-box modeling. We demonstrate the effectiveness of Split-as-a-Pro by developing a distributed algorithm for solving finite-horizon linear quadratic control problems and illustrate its use in predictive control. Our numerical case studies show that algorithms obtained using Split-as-a-Pro significantly outperform their centralized counterparts in runtime and scalability across various standard graph topologies, while seamlessly leveraging both model-based and data-driven system representations.**

I. Introduction

Control across scales is one of the grand challenges in our field [1]. Closing this gap is essential for controlling increasingly complex systems, like future energy grids, traffic networks and supply chains. While the literature on the subject is vast and opinions differ on the best approach, there is broad consensus that effective solutions should exploit system structure, integrate prior knowledge when available, and combine model-based with data-driven control — particularly when some subsystems are well-modeled and others are not.

The paper introduces operator **Split**ting **a**nd **s**calable **a**lternating **Pro**jections (Split-as-a-Pro), a new control framework that merges non-parametric system modeling with advanced optimization tools, offering a tractable and principled way to address complexity. We base our approach on behavioral systems theory [2], which models dynamical systems as sets of trajectories, independent of specific representations. The key idea is to reduce dynamic optimization problems to efficient *projection* computations. Leveraging operator splitting methods [3, 4] and alternating projection algorithms [5], we decouple constraints imposed by individual subsystems from external ones, such as interconnection and input/output constraints, allowing the projection to be computed efficiently. Split-as-a-Pro offers several advantages. First, non-parametric modeling based on behavioral system theory captures *any* system representation as a special case and makes no distinction between inputs and outputs. This provides a rigorous foundation for networked control with gray-box models that leverage both model-based and data-driven representations. Second, recognizing projections as the key bottleneck enables a systematic separation of constraints, making it possible to exploit structure, parallelize computations, and design scalable control algorithms. Third, Split-as-a-Pro does not rely on a system being linear or finite-dimensional, nor on the cost function of being quadratic, thus allowing for broad generalizations to be explored. To illustrate our findings, we revisit the finite-horizon Linear Quadratic Tracking (LQT) problem [6], a textbook example in control theory, yet one whose scalability still drives active research [7, 8].

Related work. Behavioral systems theory [2] has gained renewed attention with the rise of data-driven control, largely due to Willems' fundamental lemma [9]. The lemma enables various data-driven control frameworks [10 – 12] and, over finite time horizons, solutions to the LQT problem [13], which underpin recent direct predictive control algorithms [14 – 16]. The scalability of the LQT problem has been widely studied for state-space systems [7], as it forms the backbone of Model Predictive Control (MPC) [17]. Recent work has focused on enforcing locality constraints [8] through the System Level Synthesis (SLS) parameterization [18], and on decoupling network behavior from individual subsystems using data-driven representations to enforce dissipativity properties [19, 20]. In contrast, we develop a scalable framework that supports both model-based and data-driven representations.

Contributions. The paper offers two main contributions. First, we introduce Split-as-a-Pro, a new control framework that combines behavioral systems theory, operator splitting methods, and alternating projection algorithms. Second, we present two algorithms that showcase the Split-as-a-Pro framework allowing one to scale efficiently the solution of the LQT problem to large systems, while seamlessly integrating model-based and data-driven representations, as well as handling constraints on inputs and outputs.

Paper organization. Section II introduces notation, terminology, and preliminary results. Section III recalls the finite-horizon LQT problem, which serves as a motivating example throughout the paper. Section IV illustrates how the Split-as-a-Pro framework solves the LQT problem by distributing computations efficiently and incorporating additional constraints. Section V presents numerical case studies that demonstrate our main results. Section VI concludes with a summary and an outlook on future research directions.

Y. Tang, C. Cenedese, A. Rimoldi, Florian Dörfler, John Lygeros, are with the Department of Information Technology and Electrical Engineering at ETH Zürich (e-mails: yutang@student.ethz.ch, {ccenedese, arimoldi, dorfler, lygeros}@control.ee.ethz.ch). C. Cenedese is also with the Delft Center for Systems and Control at TU Delft. A. Padoan is with the Department of Electrical and Computer Engineering, University of British Columbia (e-mail: apadoan@ece.ubc.ca). This work was supported as a part of NCCR Automation, a National Centre of Competence in Research, funded by the Swiss National Science Foundation (grant number 51NF40_225155).

II. PRELIMINARIES

A. Notation and terminology

The set of positive integer numbers is denoted by $\mathbb{N}$. The set of real and non-negative real numbers are denoted by $\mathbb{R}$ and $\mathbb{R}_+$, respectively. For $T \in \mathbb{N}$, the set of integers $\{1, 2, \dots, T\}$ is denoted by $\mathbf{T}$. A map f from X to Y is denoted by $f : X \to Y$; $(Y)^X$ denotes the set of all such maps. The *restriction* of $f : X \to Y$ to a set X', with $X' \cap X \neq \varnothing$, is denoted by $f|_{X'}$ and is defined by $f|_{X'}(x)$ for $x \in X'$; if $\mathcal{F} \subseteq (Y)^X$, then $\mathcal{F}|_{X'}$ is defined as $\{f|_{X'} : f \in \mathcal{F}\}$. Notation for convex analysis is mostly borrowed from [4]. The gradient of a differentiable function f is denoted by ∇f. The subdifferential of a proper function f is denoted by ∂f. The proximal operator of a Convex, Closed, and Proper (CCP) function f is denoted by $\mathbf{prox}_f$ and defined as in [4, Sec. 1.3.4]. The indicator function of the set C is denoted by ι_C and the projection onto the set by P_C. The identity operator is denoted by id. The set of zeros of an operator T is denoted by $\mathbf{zer}(T)$.

B. Behavioral systems theory

1) Sequences, shift operator, and Hankel matrices. A *sequence* is a function $w : S \to \mathbb{R}^q$, also denoted by $\{w_k\}_{k \in S}$, with S a (non-empty) subset of consecutive elements of $\mathbb{N}$; the sequence w is *finite* if S is finite, *infinite* otherwise. We use the terms *sequence* and *trajectory* interchangeably. By convention, a finite sequence $w \in (\mathbb{R}^q)^{\mathbf{T}}$ is often identified with the column vector $w = \mathrm{col}(w(1), \dots, w(T)) \in \mathbb{R}^{qT}$.

The *concatenation* of sequences w and v is denoted by $w \wedge v$ and defined as in [21], with $w \wedge v = \mathrm{col}(w, v)$ if w and v are finite sequences identified with column vectors.

The *shift operator* $\sigma : (\mathbb{R}^q)^{\mathbb{N}} \to (\mathbb{R}^q)^{\mathbb{N}}$ is defined as $\sigma w(t) = w(t+1)$. By convention, the shift operator acts on all elements of a set $\mathcal{W} \subseteq (\mathbb{R}^q)^{\mathbb{N}}$, that is, $\sigma\mathcal{W} = \{\sigma w : w \in \mathcal{W}\}$.

Given a permutation matrix $\Pi \in \mathbb{R}^{q \times q}$ and an integer $0 < m < q$, the map defined by the equation

$$(u, y) = \Pi w \tag{1}$$

induces a *partition* of each $w \in \mathbb{R}^q$ into the variables $u \in \mathbb{R}^m$ and $y \in \mathbb{R}^{q-m}$. We write $w \sim (u, y)$ if (1) holds for some permutation matrix $\Pi \in \mathbb{R}^{q \times q}$ and integer $0 < m < q$. Any partition of the form (1) induces natural projections $\pi_u : w \mapsto u$ and $\pi_y : w \mapsto y$.

The *Hankel matrix* of depth $L \in \mathbf{T}$ associated with the sequence $w \in (\mathbb{R}^q)^{\mathbf{T}}$ is defined as

$$H_L(w) = \begin{bmatrix} w(1) & w(2) & \cdots & w(T-L+1) \\ w(2) & w(3) & \cdots & w(T-L+2) \\ \vdots & \vdots & \ddots & \vdots \\ w(L) & w(L+1) & \cdots & w(T) \end{bmatrix}.$$

2) Systems. In behavioral systems theory [2], a *system* is a triple $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$, where $\mathbb{T}$ is the *time set*, $\mathbb{W}$ is the *signal set*, and $\mathcal{B} \subseteq (\mathbb{W})^{\mathbb{T}}$ is the *behavior* of the system. Throughout this work, we exclusively focus on *discrete-time* systems, with $\mathbb{T} = \mathbb{N}$ and $\mathbb{W} = \mathbb{R}^q$. We adapt definitions accordingly, emphasizing this assumption when necessary. We also identify each system Σ with the corresponding behavior $\mathcal{B}$.

3) Linear Time-Invariant (LTI) systems. A system $\mathcal{B}$ is *linear* if $\mathcal{B}$ is a linear subspace, *time-invariant* if $\mathcal{B}$ is shift-invariant, *i.e.*, $\sigma(\mathcal{B}) \subseteq \mathcal{B}$, and *complete* if $\mathcal{B}$ is closed in the topology of pointwise convergence [22]. The set of all (discrete-time) complete LTI systems is denoted by $\mathcal{L}^q$. We often simply write $\mathcal{B} \in \mathcal{L}^q$.

4) Kernel representations. Every LTI system $\mathcal{B} \in \mathcal{L}^q$ admits a *kernel representation* of the form $\mathcal{B} = \ker R(\sigma)$, where $R(\sigma)$ is the operator defined by the polynomial matrix $R(z) = R_0 + R_1 z + \ldots + R_\ell z^\ell$, with $R_i \in \mathbb{R}^{p \times q}$ for $i \in \boldsymbol{\ell}$, and $\ker R(\sigma) = \{w \in (\mathbb{R}^q)^{\mathbb{N}} : R(\sigma)w = 0\}$. Without loss of generality, we assume that $\ker R(\sigma)$ is a *minimal* kernel representation of $\mathcal{B}$, *i.e.*, $R(\sigma)$ has full row rank [23].

5) Integer invariants. The structure of an LTI system $\mathcal{B} \in \mathcal{L}^q$ is characterized by a set of *integer invariants* [22], defined as

- the *number of inputs* $m(\mathcal{B}) = q - \text{row dim} R(\sigma)$,
- the *number of outputs* $p(\mathcal{B}) = \text{row dim} R(\sigma)$,
- the *lag* $\ell(\mathcal{B}) = \max_{i \in \mathbf{p}} \{\deg \text{row}_i R(\sigma)\}$, and
- the *order* $n(\mathcal{B}) = \sum_{i \in \mathbf{p}} \deg \text{row}_i R(\sigma)$,

where $\ker R(\sigma)$ is a minimal kernel representation of $\mathcal{B}$, while row dim$R$ and $\deg \text{row}_i R$ are the number of rows and the degree of the i-th row of $R(z)$, respectively. The integer invariants are intrinsic properties of a system, as they do not depend on its representation [22]. Thus, we omit the dependence on the particular behavior if clear from context.

6) State-space representations. Every LTI system $\mathcal{B} \in \mathcal{L}^q$ can be described by the equations

$$\sigma x = Ax + Bu, \quad y = Cx + Du, \tag{2}$$

and admits a (*minimal*) *input/state/output representation*

$$\mathcal{B} = \left\{(u, y) \sim w \in (\mathbb{R}^q)^{\mathbb{N}} : \exists\, x \in (\mathbb{R}^n)^{\mathbb{N}} \text{ s.t. (2) holds}\right\}, \tag{3}$$

where $\left[\begin{smallmatrix} A & B \\ C & D \end{smallmatrix}\right] \in \mathbb{R}^{(n+p) \times (n+m)}$ and m, n, and p are the number of inputs, the order, and the number of outputs, respectively.

7) Data-driven representations. Data-driven representations describe system behaviors using raw data matrices, leveraging the fact that, over a finite time horizon, the behavior of an LTI system is a subspace with dimension defined by the integer invariants and time horizon. Next, we summarize a version of this principle known as the *fundamental lemma* [9].

Lemma 1. [21] Let $\mathcal{B} \in \mathcal{L}^q$ and $w \in \mathcal{B}|_{[1,T]}$. Fix $L \in \mathbf{T}$, with $L > \ell$. Then $\mathcal{B}|_{[1,L]} = \mathbf{im}\, H_L(w)$ if and only if

$$\mathbf{rank}\, H_L(w) = mL + n. \tag{4}$$

The rank condition (4) is referred to as the *generalized persistency of excitation* condition [21]. Different variations of this principle can be formulated under a range of assumptions, see, *e.g.*, the survey [21] for an overview and [24, 25] for some recent extensions.

III. Problem formulation

We consider the finite-horizon LQT problem — a textbook example of control design [6]. Despite its simplicity, this example highlights the generality and tractability of our framework in addressing fundamental control problems, the scalability of which is nontrivial and still an active area of research both in model-based and data-driven contexts [7, 8].

Given an LTI system $\mathcal{B} \in \mathcal{L}^q$, the goal of LQT is to find a trajectory $w_{\mathrm{f}}^\star \in \mathcal{B}$ that is as close as possible to a given reference trajectory w_{ref}. Over a finite-horizon, the problem can be formulated as that of minimizing the quadratic cost [13]

$$\|w - w_{\mathrm{ref}}\|^2_{I\otimes\Phi} = \sum_{t=1}^{T_{\mathrm{f}}} (w(t) - w_{\mathrm{ref}}(t))^{\mathsf{T}} \Phi (w(t) - w_{\mathrm{ref}}(t)), \quad (5)$$

where $\Phi \in \mathbb{R}^{q\times q}$ is a given symmetric positive definite matrix.

Problem 1 (Finite-horizon LQT)**.** Given an LTI system $\mathcal{B} \in \mathcal{L}^q$, a reference trajectory $w_{\mathrm{ref}} \in \mathbb{R}^{qT_{\mathrm{f}}}$, an initial trajectory $w_{\mathrm{ini}} \in \mathcal{B}|_{[1,T_{\mathrm{ini}}]}$, and a symmetric positive definite matrix $\Phi \in \mathbb{R}^{q\times q}$, find a trajectory $w_{\mathrm{f}}^\star \in \mathcal{B}|_{[1,T_{\mathrm{f}}]}$ that minimizes the quadratic cost (5) and has w_{ini} as a prefix trajectory, *i.e.*,

$$\begin{aligned} \min_{w_{\mathrm{f}}\in\mathbb{R}^{qT_{\mathrm{f}}}} \quad & \|w_{\mathrm{f}} - w_{\mathrm{ref}}\|^2_{I\otimes\Phi} \\ \text{s.t.} \quad & w_{\mathrm{ini}} \wedge w_{\mathrm{f}} \in \mathcal{B}|_{[1,T_{\mathrm{ini}}+T_{\mathrm{f}}]}. \end{aligned} \quad (6)$$

For $T_{\mathrm{ini}} \geqslant \ell$, the prefix trajectory w_{ini} implicitly fixes the initial condition for the optimal control problem (6), leading to a unique solution $w_{\mathrm{f}}^\star$ that can be explicitly calculated [13]. This reduces to solving a constrained least squares problem, requiring $T_{\mathrm{f}}(m+p+n)(m+n)^2$ operations [26, p.368]. Complexity grows only linearly in the time horizon T_{f}, but cubically in the order of the system n and the number of inputs m. Moreover, this formulation does not readily integrate different system representations or exploit prior knowledge (e.g., interconnection topology). To address these limitations, we leverage the Split-as-a-Pro framework, exploiting system structure via operator splitting methods [4] and alternating projection algorithms [5].

Remark 1 (Constraints)**.** While not part of the original LQT problem, constraints are crucial for MPC applications and recent data-driven variants [14 – 16], which solve the finite-horizon LQT problem in a receding-horizon fashion. Constraints can easily be incorporated into the LQT problem (6) by enforcing the additional constraint $w_{\mathrm{f}} \in \mathcal{C}$, for a suitable choice of $\mathcal{C}$. As discussed in Section IV-C and illustrated numerically in Section V, constraints are readily handled by the Split-as-a-Pro framework, provided that the projection onto $\mathcal{C}$ is "simple," as is the case of box, non-negativity, or half-space constraints arising in predictive control [27]. △

IV. Main results

The Split-as-a-Pro framework takes advantage of structural properties to efficiently solve dynamic optimization problems (e.g., control and filtering) using behavioral, non-parametric system representations. Split-as-a-Pro involves three key steps summarized next and developed further individually in the remainder of the section.

A. *Monotone inclusion problem*: Reformulate the original optimization problem as a monotone inclusion problem, possibly in a higher-dimensional space.

B. *Operator splitting*: Select an operator splitting method to derive an iterative algorithm that solves the monotone inclusion problem, with convergence guarantees tied to the specific splitting method. The main idea is to make the primary computational bottleneck the *projection* onto the intersection of multiple closed, convex sets.

C. *Alternating projections*: Decompose the projection task using an alternating projection algorithm into simpler projections onto individual sets, whose intersection forms the target set. This step reduces computational load by replacing a single complex projection with efficient, parallelizable projections, isolating subsystem-specific constraints from interconnection or dynamic constraints.

Next, we apply this framework to the LQT problem, showing how it enables the design of scalable algorithms that leverage non-parametric, finite-horizon system representations, compatible with both model-based and data-driven approaches.

A. Monotone inclusion problem

The first step in our analysis is to reformulate problem (6) as a monotone inclusion problem. To illustrate the flexibility of Split-as-a-Pro, we present two possible reformulations.

1) Monotone inclusion with two operators. The first is given by the (higher-dimensional) optimization problem

$$\min_{w\in\mathbb{R}^{q(T_{\mathrm{ini}}+T_{\mathrm{f}})}} \; g(w) + h(w), \quad (7)$$

where g and h are extended real-valued functions, defined as

$$g(w) = \iota_{\mathcal{B}|_{[1,T_{\mathrm{ini}}+T_{\mathrm{f}}]}}(w) + \iota_{\{w_{\mathrm{ini}}\}}(\pi_{\mathrm{ini}}(w)), \quad (8a)$$

$$h(w) = \|\pi_{\mathrm{f}}(w) - w_{\mathrm{ref}}\|^2_{I\otimes\Phi}, \quad (8b)$$

where the projections $\pi_{\mathrm{ini}} : w \mapsto \Pi_{\mathrm{ini}} w$ and $\pi_{\mathrm{f}} : w \mapsto \Pi_{\mathrm{f}} w$ are defined by the matrices $\Pi_{\mathrm{ini}} = [I_{qT_{\mathrm{ini}}} \; 0_{qT_{\mathrm{ini}}\times qT_{\mathrm{f}}}]$ and $\Pi_{\mathrm{f}} = [0_{qT_{\mathrm{f}}\times qT_{\mathrm{ini}}} \; I_{qT_{\mathrm{f}}}]$, respectively.

This reformulation replaces the optimization variable w_{f} with the higher-dimensional variable w, constrained to have a fixed prefix w_{ini}. The constraints $w \in \mathcal{B}|_{[1,T_{\mathrm{ini}}+T_{\mathrm{f}}]}$ and $\pi_{\mathrm{ini}}(w) = w_{\mathrm{ini}}$ are lifted into the objective via indicator functions. The corresponding monotone inclusion problem is

$$0 \ni \partial g + \nabla h. \quad (9)$$

Fact 1. Consider the optimization problem (6) and the monotone inclusion problem (9), with g and h defined as in (8). Assume $\Phi \in \mathbb{R}^{q\times q}$ is symmetric and positive definite, $w_{\mathrm{ini}} \in \mathcal{B}|_{[1,T_{\mathrm{ini}}]}$, and $T_{\mathrm{ini}} \geqslant \ell$. Then $w^\star \in \mathbf{zer}(\partial g + \nabla h)$ if and only if $\pi_{\mathrm{f}}(w^\star)$ is a minimizer of (6).

Various reformulations can be tailored to use different operator splitting methods that take advantage of the particular properties of the terms appearing in the inclusion. An alternative reformulation with three operators better suits scalability requirements and accommodates additional constraints, such as those arising in predictive control, as discussed next.

2) Monotone inclusion with three operators. Another possible equivalent reformulation of problem (6) is given by the optimization problem

$$\min_{w \in \mathbb{R}^{q(T_{\text{ini}}+T_{\text{f}})}} \; f(w) + g(w) + h(w), \tag{10}$$

where f, g, and h extended real-valued functions, defined as

$$f(w) = \iota_{\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}}(w), \tag{11a}$$
$$g(w) = \iota_{\{w_{\text{ini}}\}}(\pi_{\text{ini}}(w)), \tag{11b}$$
$$h(w) = \|\pi_{\text{f}}(w) - w_{\text{ref}}\|^2_{I\otimes\Phi}. \tag{11c}$$

This reformulation mirrors the previous one, but differs by explicitly separating the constraints $w \in \mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ and $\pi_{\text{ini}}(w) = w_{\text{ini}}$, enabling distributed algorithms that scale efficiently and naturally accommodate additional constraints. The corresponding monotone inclusion problem is

$$0 \ni \partial f + \partial g + \nabla h. \tag{12}$$

Fact 2. Consider the optimization problem (6) and the monotone inclusion problem (12), with g and h defined as in (11). Assume $\Phi \in \mathbb{R}^{q\times q}$ is symmetric and positive definite, $w_{\text{ini}} \in \mathcal{B}|_{[1,T_{\text{ini}}]}$, and $T_{\text{ini}} \geqslant \ell$. Then $w^\star \in \mathbf{zer}(\partial f + \partial g + \nabla h)$ if and only if $\pi_{\text{f}}(w^\star)$ is a minimizer of (6).

B. Operator splitting

The second step in our framework is to solve the monotone inclusion problem through a fixed-point iteration of the form

$$w_{k+1} = T(w_k), \quad k \in \mathbb{N}, \tag{13}$$

with T an operator defined by a given splitting method [4].

1) Forward-Backward (FB) splitting. One of the most well-known splitting methods is the Forward-Backward splitting [4, p.46]. Since g and h are CCP functions, with h differentiable, this method is applicable to problem (7). The solution is defined by the fixed-point iteration (13), with T defined as

$$T = \mathbf{prox}_{\alpha\partial g}\left(\mathrm{id} - \alpha\nabla h\right). \tag{14}$$

Since g is the indicator function of the affine set

$$\mathcal{A}|_{[1,T_{\text{ini}}+T_{\text{f}}]} = \{w \in \mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]} \mid w|_{[1,T_{\text{ini}}]} = w_{\text{ini}}\}, \tag{15}$$

this yields a standard *projected gradient algorithm* [4, p.49], described in pseudo-code in Algorithm 1.

Algorithm 1

Input: Initial trajectory $w_{\text{ini}} \in \mathbb{R}^{qT_{\text{ini}}}$, reference trajectory $w_{\text{ref}} \in \mathbb{R}^{qT_{\text{f}}}$, weight matrix $\Phi \in \mathbb{R}^{q\times q}$, parameter $\alpha \in \mathbb{R}_+$, initial guess $w_1 \in \mathbb{R}^{q(T_{\text{f}}+T_{\text{ini}})}$.
Output: Sequence of iterates $\{w_k\}_{k\in\mathbb{N}}$.

for $k = 1, 2, \ldots$ **do**
  $z_{k+1} = 2\Pi_{\text{f}}{}^{\mathsf{T}}(I \otimes \Phi)(\Pi_{\text{f}} w_k - w_{\text{ref}})$
  $w_{k+1} = P_{\mathcal{A}|_{[1,T_{\text{ini}}+T_{\text{f}}]}}\left(w_k - \alpha z_{k+1}\right)$
end for

The next statement gives conditions for Algorithm 1 to yield the desired solution. The proof is deferred to the appendix.

Theorem 1 (Convergence of Algorithm 1)**.** Consider the optimization problem (7), with g and h defined as in (8). Let $\Phi \in \mathbb{R}^{q\times q}$ be symmetric and positive definite, $w_{\text{ini}} \in \mathcal{B}|_{[1,T_{\text{ini}}]}$, with $T_{\text{ini}} \geqslant \ell$, and $\alpha \in (0, 1/\rho(\Phi))$. Then the sequence $\{w_k\}_{k\in\mathbb{N}}$ generated by Algorithm 1 converges to the unique global optimum of the optimization problem (7).

2) Davis-Yin (DY) Splitting. Another particularly useful splitting for our purposes is the DY splitting [4, p.48]. Since f, g, and h are CCP functions, with h differentiable, this method applies to problem (10). The solution is defined by the fixed-point iteration (13), with T defined as

$$T = \mathrm{id} - \mathbf{prox}_{\alpha\partial g} + \mathbf{prox}_{\alpha\partial f}\left(2\mathbf{prox}_{\alpha\partial g} - \mathrm{id} - \alpha\nabla h \circ \mathbf{prox}_{\alpha\partial g}\right). \tag{16}$$

Since f and g are indicator functions of specific sets — the subspace $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ and the singleton $\{w_{\text{ini}}\}$ — the proximal operator of their subdifferentials reduces to a projection onto such sets [4, p.42]. The resulting iterative algorithm is described in pseudocode in Algorithm 2.

Algorithm 2

Input: Initial trajectory $w_{\text{ini}} \in \mathbb{R}^{qT_{\text{ini}}}$, reference trajectory $w_{\text{ref}} \in \mathbb{R}^{qT_{\text{f}}}$, weight matrix $\Phi \in \mathbb{R}^{q\times q}$, parameter $\alpha \in \mathbb{R}_+$, initial guess $w_1 \in \mathbb{R}^{q(T_{\text{f}}+T_{\text{ini}})}$.
Output: Sequence of iterates $\{w_k\}_{k\in\mathbb{N}}$.

for $k = 1, 2, \ldots$ **do**
  $z_{k+1/2} = w_{\text{ini}} \wedge \Pi_{\text{f}} w_k$
  $z_{k+1} = 2\Pi_{\text{f}}{}^{\mathsf{T}}(I \otimes \Phi)(\Pi_{\text{f}} z_{k+1/2} - w_{\text{ref}})$
  $v_{k+1} = P_{\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}}\left(2z_{k+1/2} - w_k - \alpha z_{k+1}\right)$
  $w_{k+1} = w_k + v_{k+1} - z_{k+1/2}$
end for

Next, we provide conditions for Algorithm 2 to produce the desired solution. The proof can be found in the appendix.

Theorem 2 (Convergence of Algorithm 2)**.** Consider the optimization problem (10), with f, g, and h defined as in (11). Let $\Phi \in \mathbb{R}^{q\times q}$ be symmetric and positive definite, $w_{\text{ini}} \in \mathcal{B}|_{[1,T_{\text{ini}}]}$, with $T_{\text{ini}} \geqslant \ell$, and $\alpha \in (0, 1/\rho(\Phi))$. Then the sequence $\{w_k\}_{k\in\mathbb{N}}$ generated by Algorithm 2 converges to the unique global optimum of the optimization problem (10).

By design, Algorithms 1 and 2 yield the desired solution to the associated problems. All steps are highly parallelizable and conducive to scalable implementation, particularly when the cost is separable (e.g., Φ block-diagonal) [27]. The only exception is the projection onto the sets $\mathcal{A}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ and $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$, which poses a fundamental scalability challenge. For an LTI system $\mathcal{B} \in \mathcal{L}^q$, the projection can be efficiently computed either in closed form or by solving a linear system in saddle point form [28]. If $B \in \mathbb{R}^{q(T_{\text{ini}}+T_{\text{f}})\times r}$ is a matrix with orthonormal columns spanning the subspace $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$, the projection is given by

$$P_{\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}}(w) = B(B^{\mathsf{T}}B)^{-1}B^{\mathsf{T}}w.$$

Thus, computing the projection $P_{\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}}$ requires inverting a $r \times r$ matrix and $2r^3/3$ operations [26, p.210]. Since

$r = \dim \mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]} = m(T_{\text{ini}} + T_{\text{f}}) + n$ [9], the complexity grows cubically with the number of inputs m and the system order n, posing a challenge in predictive control applications, where projections must be computed repeatedly in a receding-horizon fashion. In such settings, the projection matrix can be precomputed offline, reducing each projection to a matrix-vector multiplication costing $q^2(T_{\text{ini}} + T_{\text{f}})^2$ operations [29, p.4]. Yet, the offline computation can be prohibitively expensive—or even impossible—when dealing with large-scale systems with subsystems that are accessible only through local input-output data. Similar complexity considerations apply to the projection onto the affine set $\mathcal{A}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$. To address these limitations, we use alternating projections to exploit system structure, distribute projection computations, and support the simultaneous use of model-based and data-driven system representations, as discussed next.

C. Alternating projections

The third step in our analysis leverages alternating projection algorithms to exploit system structure.

1) Alternating projection algorithms. Alternating projection algorithms aim to find a point in the intersection of two sets [5]. A classical alternating projection algorithm, introduced by von Neumann [5], is defined by the iteration

$$w_{k+1} = P_{\mathcal{C}_2}\left(P_{\mathcal{C}_1}\left(w_k\right)\right), \quad k \in \mathbb{N}, \tag{17}$$

where $\mathcal{C}_1$ and $\mathcal{C}_2$ are closed subspaces of a Hilbert space with non-empty intersection. The main advantage is that, while projecting onto the intersection may be costly, projections onto individual sets are often efficient. This idea generalizes to more than two sets. A well-known extension for multiple sets is Dykstra's algorithm [3, p.556], defined by the iteration

$$w_{k+1} = P_{\mathcal{C}_s}\left(P_{\mathcal{C}_{s-1}}\left(\cdots P_{\mathcal{C}_1}\left(w_k\right)\cdots\right)\right), \quad k \in \mathbb{N}. \tag{18}$$

where $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_s$, are closed convex subsets of a Hilbert space with non-empty intersection.

Alternating projection algorithms can be used to replace the monolithic projection steps in Algorithms 1 and 2. The key idea is to decompose the sets $\mathcal{A}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ and $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ into sets onto which projections can be computed efficiently. For example, this applies when $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ arises from an interconnected LTI system, which one can model as the intersection of the set of all isolated subsystem behaviors $\mathcal{C}_1$ with the set $\mathcal{C}_2$ of all interconnection constraints [19 , 20]. The decomposition simplifies the projection step and lends itself to parallel implementation, which is advantageous in large-scale settings. In Algorithm 1, this replaces the projection onto the affine set (15) with projections onto three sets: two sets $\mathcal{C}_1$ and $\mathcal{C}_2$ whose intersection define $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ together with the affine set defined by the constraint $\pi_{\text{ini}}(w) = w_{\text{ini}}$. In Algorithm 2, the projection onto $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ is similarly replaced by alternating projections onto $\mathcal{C}_1$ and $\mathcal{C}_2$.

The alternating projection framework also extends naturally to constraints not originally part of the LQT problem. For instance, additional constraints can be incorporated by introducing convex sets $\mathcal{C}_3, \ldots, \mathcal{C}_s$ and applying Dykstra's iteration (18) over $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_s$. As noted in Remark 1, such constraints are standard in MPC and recent data-driven variants, and often include box or non-negativity constraints (e.g., for saturation or physical limits). These constraints fit naturally within the Split-as-a-Pro framework, as the required projections are straightforward to compute, e.g., for standard sets like box, non-negativity, or half-space constraints [27, Section 6]. Section V provides numerical illustrations of this extension in the context of data-driven predictive control.

2) LQT with Split-as-a-Pro. Returning to the LQT problem, we now illustrate how to replace single-step projections with alternating projection algorithms. For illustration, we consider Algorithm 2 and substitute the projection in Step 4 with von Neumann's iteration (17). This yields a distributed version of Algorithm 2, whose pseudocode is given in Algorithm 3.

Algorithm 3

Input: Initial trajectory $w_{\text{ini}} \in \mathbb{R}^{qT_{\text{ini}}}$, reference trajectory $w_{\text{ref}} \in \mathbb{R}^{qT_{\text{f}}}$, weight matrix $\Phi \in \mathbb{R}^{q\times q}$, parameters $\alpha \in \mathbb{R}_+$, number of inner loop iterations $J \in \mathbb{N}$, initial guess $w_1 \in \mathbb{R}^{q(T_{\text{f}}+T_{\text{ini}})}$.

Output: Sequence of iterates $\{w_k\}_{k\in\mathbb{N}}$.

for $k = 1, 2, \ldots,$ **do**
  $z_{k+1/2} = w_{\text{ini}} \wedge \Pi_{\text{f}} w_k$
  $z_{k+1} = \Pi_{\text{f}}{}^{\mathsf{T}}(I \otimes \Phi)(\Pi_{\text{f}} z_{k+1/2} - w_{\text{ref}})$
  $v_{k+1,0} = 2z_{k+1} - w_k - 2\alpha z_{k+1/2}$
  for $j = 0, 1, 2, \ldots, J-1$ **do**
    $v_{k+1,j+1/2} = P_{\mathcal{C}_1}\left(v_{k+1,j}\right)$
    $v_{k+1,j+1} = P_{\mathcal{C}_2}\left(v_{k+1,j+1/2}\right)$
  end for
  $w_{k+1} = w_k + v_{k+1,J} - z_{k+1/2}$
end for

The inner loop of Algorithm 3 performs J iterations, each involving two projections. If projections onto the individual sets are efficient, with worst-case per-iteration cost $O(r)$, the total cost scales as $O(Jr)$. This can offer significant advantages when the interconnection structure is sparse and projections decompose well. A detailed analysis of how to exploit such structure and implement the associated projections efficiently is deferred to future work. As previously discussed, the alternating projection step can accommodate additional constraints by introducing further sets $\mathcal{C}_3, \mathcal{C}_4, \ldots, \mathcal{C}_n$ and applying Dykstra's iteration (18). Section V presents empirical evidence that Algorithm 3 outperforms centralized methods in sparse interconnection settings and, despite its simplicity, yields control performance on par with state-of-the-art data-driven predictive control algorithms.

Remark 2 (Early termination)**.** Residual errors due to early termination of the inner loop are inevitable in Algorithm 3. Thus, it is crucial to determine whether Algorithm 2 converges after a minimum number of alternating projection iterations while accounting for these errors. With a judicious choice of inner loop iterations J, convergence can be established using Fejér monotonicity [3] or inexact operator splitting methods [4]. The detailed proof of convergence is deferred to future work due to space constraints. △

V. Numerical case studies

We now illustrate the scalability of Split-as-a-Pro by comparing the (centralized) Algorithm 2 and the (distributed) Algorithm 3 on three common graph topologies — chain, ring, and lattice — by increasing the number of subsystems ν and analyzing runtime performance on a fixed LQT problem.

1) Scalable control for interconnected system. We consider an interconnected system inspired by [18], which consists of $\nu \in \mathbb{N}$ subsystems described by the state-space representations

$$\begin{aligned} x_i(t+1) &= A_i x_i(t) + B_i u_i(t) + \sum_{j \neq i} B_i u_{i,j}(t), \\ y_i(t) &= C_i x_i(t), \end{aligned} \tag{19}$$

where $x_i(t) \in \mathbb{R}^2$, $u_i(t) \in \mathbb{R}$, $u_{i,j}(t) \in \mathbb{R}$, $y_i(t) \in \mathbb{R}$, and

$$A_i = \begin{bmatrix} 1 & \Delta t \\ -\frac{K_i}{m_i}\Delta t & 1 - \frac{d_i}{m_i}\Delta t \end{bmatrix}, B_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C_i = \begin{bmatrix} \frac{k_i}{m_i}\Delta t & 0 \end{bmatrix},$$

with $m_i > 0$, $d_i > 0$, $k_i > 0$, $K_i > 0$, and $\Delta t > 0$. To illustrate compatibility with gray-box modeling, we model half of the subsystems using state-space representations, while the other half are represented by Hankel matrices derived from suitably collected data [21]. The parameters of each subsystem are sampled from a uniform distribution. We select $T_{\text{ini}} = 2\nu + 1$ and $T_{\text{f}} = 5$. We collect input-output data $w_{\text{d}} \sim (u_{\text{d}}, y_{\text{d}})$ by applying pseudo-random inputs from a uniform distribution in $[-1, 1]$, with $T = \nu(T_{\text{ini}} + T_{\text{f}}) + 200$ samples. The parameters in Algorithms 1, 2 and 3 are selected as $\alpha = 0.1$ and $\Phi = I$, respectively. For Algorithm 3, we set $J = 5$, which empirically generates satisfactory controller performance. Figure 1 compares the mean runtime of Algorithms 1, 2 and 3 across the three graph topologies averaged over five different runs. For all three graph topologies, Algorithms 1 and 2 are outperformed by Algorithm 3 even for moderately large-scale systems.

2) Beyond linear quadratic tracking. The Split-as-a-Pro framework extends well beyond LQT problems — for instance, to input-constrained problems. For illustration, we use a variant of Algorithm 3, which includes an additional projection step onto the input constraint set, to solve the Data-driven Model Predictive Control (DD-MPC) problem from [16] in a receding horizon fashion. We consider the LTI system described above with $\nu = 2$ subsystems arranged in a chain graph topology. The reference is selected as $w_{\text{ref}} \sim (u_{\text{ref}}, y_{\text{ref}})$, with $u_{\text{ref}} = 0.25$ and $y_{\text{ref}} = 0.25$. The cost is defined by the block diagonal matrix $\Phi = \text{diag}(0.1I, 10I)$. We add input box constraints $u_i(t) \in [-0.5, 0.5]$. Thus, we leverage von Neumanns's alternating projection algorithm (17), with $\mathcal{C}_1 = \mathcal{B}|_{[1, T_{\text{ini}}+T_{\text{f}}]}$ and $\mathcal{C}_2 = [-0.5, 0.5]^{m(T_{\text{ini}}+T_{\text{f}})}$. We also add a disturbance to the system at the time $t = 50$. Fig. 2 shows the input/output trajectory of the first subsystem obtained using the DD-MPC algorithm given in [16] with a Quadratic Programming (QP) solver (solid) and Split-as-a-Pro (dashdotted), together with reference values (dashed) and constraints (shaded). The results show that Split-as-a-Pro attains control performance on par with those obtained using a QP solver, while effectively enforcing input constraints and adapting to exogenous disturbances.

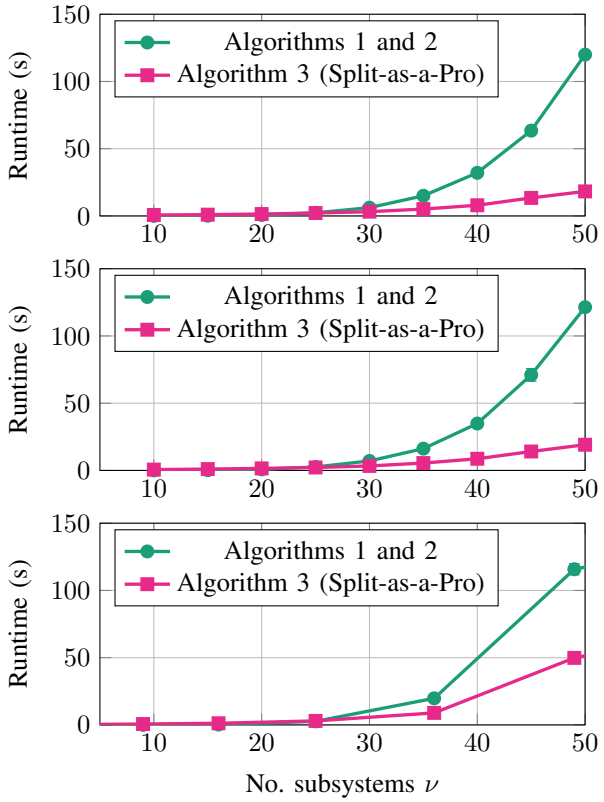

Fig. 1: Runtime of Algorithms 1, 2 and 3 as a function of the number of interconnected subsystems ν for chain (top), ring (center), and lattice (bottom) topologies.

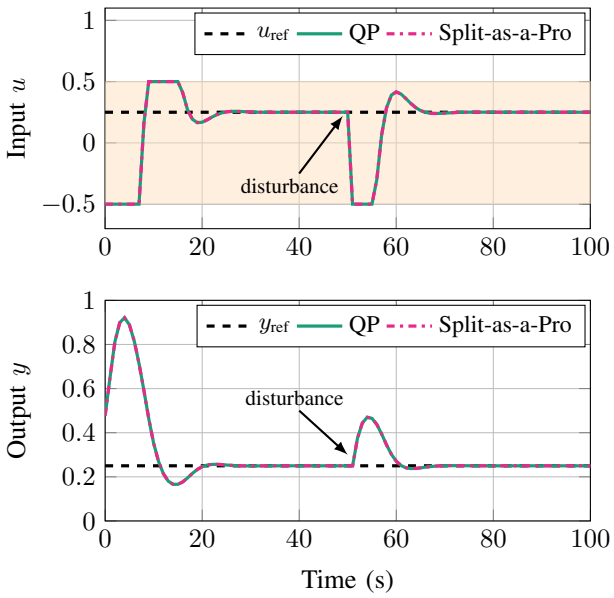

Fig. 2: Control performance of DD-MPC using a QP solver (solid) and Split-as-a-Pro (dashdotted), together with reference values (dashed) and input constraints set (shaded).

VI. Conclusion

The paper has introduced Split-as-a-Pro, a control framework that integrates behavioral systems theory with operator splitting methods and alternating projection algorithms. The framework reduces large-scale control problems to the efficient computation of a projection, offering advantages in terms of scalability and compatibility with gray box modeling. We have showcased the benefits of Split-as-a-Pro by developing a distributed algorithm for solving finite-horizon linear quadratic control problems, showing that it significantly outperforms its centralized counterparts for large-scale systems across various graph topologies and match state-of-the-art data-driven predictive control algorithms. Future research will extend the Split-as-a-Pro framework to a wider class of dynamic optimization problems.

References

[1] A. M. Annaswamy, K. H. Johansson, and G. Pappas, "Control for societal-scale challenges: Road map 2030," *IEEE Control Syst. Mag.*, vol. 44, no. 3, pp. 30–32, 2024.

[2] J. C. Willems, "The behavioral approach to open and interconnected systems," *IEEE Control Syst. Mag.*, vol. 27, no. 6, pp. 46–99, 2007.

[3] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, 2nd ed. Cham, Switzerland: Springer, 2017.

[4] E. K. Ryu and W. Yin, *Large-scale convex optimization: algorithms & analyses via monotone operators*. Cambridge, U. K.: Cambridge Univ. Press, 2023.

[5] R. Escalante and M. Raydan, *Alternating projection methods*. Philadelphia, PA, USA: SIAM, 2011.

[6] B. D. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods*. New York, NY, USA: Dover Publications, 2007.

[7] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. Syst. Tech.*, vol. 18, no. 2, pp. 267–278, 2009.

[8] J. S. Li, C. A. Alonso, and J. C. Doyle, "Frontiers in scalable distributed control: SLS, MPC, and beyond," in *Proc. Amer. Control Conf.*, New Orleans, LA, USA, 2021, pp. 2720–2725.

[9] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. M. De Moor, "A note on persistency of excitation," *Syst. Control Lett.*, vol. 54, no. 4, pp. 325–329, 2005.

[10] C. De Persis and P. Tesi, "Formulas for data-driven control: Stabilization, optimality, and robustness," *IEEE Trans. Autom. Control*, vol. 65, no. 3, pp. 909–924, 2020.

[11] H. J. van Waarde, J. Eising, H. L. Trentelman, and M. K. Camlibel, "Data informativity: A new perspective on data-driven analysis and control," *IEEE Trans. Autom. Control*, vol. 65, no. 11, pp. 4753–4768, 2020.

[12] W. Favoreel, B. De Moor, and M. Gevers, "Spc: Subspace predictive control," *IFAC Proc. Vol.*, vol. 32, no. 2, pp. 4004–4009, 1999.

[13] I. Markovsky and P. Rapisarda, "Data-driven simulation and control," *Int. J. Control*, vol. 81, no. 12, pp. 1946–1959, 2008.

[14] J. Coulson, J. Lygeros, and F. Dörfler, "Data-enabled predictive control: In the shallows of the DeePC," in *Proc. Eur. Control Conf.*, Napoli, Italy, 2019, pp. 307–312.

[15] V. Breschi, A. Chiuso, and S. Formentin, "Data-driven predictive control in a stochastic setting: a unified framework," *Automatica*, vol. 152, p. 110961, 2023.

[16] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, "Data-driven model predictive control with stability and robustness guarantees," *IEEE Trans. Autom. Control*, vol. 66, no. 4, pp. 1702–1717, 2021.

[17] J. B. Rawlings and D. Q. Mayne, *Model predictive control: Theory and design*. Madison, Wisconsin: Nob Hill Pub., 2009.

[18] J. Anderson, J. C. Doyle, S. H. Low, and N. Matni, "System level synthesis," *Ann. Rev. Control*, vol. 47, pp. 364–393, 2019.

[19] Y. Yan, J. Bao, and B. Huang, "Behavioural approach to distributed control of interconnected systems," *arXiv preprint*, 2021. [Online]. Available: https://arxiv.org/abs/2103.10063

[20] ——, "Distributed data-driven predictive control via dissipative behavior synthesis," *IEEE Trans. Autom. Control*, vol. 69, no. 5, pp. 2899–2914, 2023.

[21] I. Markovsky and F. Dörfler, "Behavioral systems theory in data-driven analysis, signal processing, and control," *Ann. Rev. Control*, vol. 52, pp. 42–64, 2021.

[22] J. C. Willems, "From time series to linear system—Part I. Finite dimensional linear time invariant systems," *Automatica*, vol. 22, no. 5, pp. 561–580, 1986.

[23] ——, "Models for dynamics," *Dynamics reported: a series in dynamical systems and their applications*, pp. 171–269, 1989.

[24] A. Padoan, F. Dörfler, and J. Lygeros, "Data-driven representations of conical, convex, and affine behaviors," in *Proc. 62nd Conf. Decision Control*, Singapore, 2023, pp. 596–601.

[25] J. Berberich, A. Iannelli, A. Padoan, J. Coulson, F. Dörfler, and F. Allgöwer, "A quantitative and constructive proof of Willems' fundamental lemma and its implications," in *Proc. Amer. Control Conf.*, San Diego, CA, USA, 2023, pp. 4155–4160.

[26] S. Boyd and L. Vandenberghe, *Introduction to applied linear algebra: vectors, matrices, and least squares*. Cambridge, U. K.: Cambridge Univ. Press, 2018.

[27] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, 2014.

[28] M. Benzi, G. H. Golub, and J. Liesen, "Numerical solution of saddle point problems," *Acta numerica*, vol. 14, pp. 1–137, 2005.

[29] G. H. Golub and C. F. Van Loan, *Matrix computations (4th Ed.)*. Baltimore, MD, USA: Johns Hopkins Univ. Press, 2013.

Appendix

A. Proofs of Theorems 1 and 2

We recall sufficient conditions for convergence of the fixed point iteration (13) with T defined as in (16) [4, Ch. 2].

Lemma 2 (Convergence of DY splitting algorithm)**.** Consider the optimization problem (10). Assume

(i) f and g are CCP functions,
(ii) h is a convex function with γ-Lipschitz gradient,
(iii) $\mathbf{zer}(\partial f + \partial g + \nabla h) \neq \varnothing$,
(iv) $\alpha \in (0, 2/\gamma)$.

Then the sequence $\{w_k\}_{k\in\mathbb{N}}$ defined by (13), with T defined as in (16), converges to some $w^\star \in \mathbf{zer}(\partial f + \partial g + \nabla h)$.

If $g = 0$ in (16) then we retrieve (14) where $\partial g := \partial f$. This lemma also implies convergence of the sequence attained from (13), with T as in (14), to $w^\star \in \mathbf{zer}(\partial g + \nabla h)$ under the same assumptions on g and h. We now prove Theorem 2; the proof of Theorem 1 is analogous and, hence, omitted.

Proof of Theorem 2: First, we show that the assumptions of Lemma 2 hold, with f, g, and h defined as in (11).

(i) If a set $C \subseteq \mathbb{R}^n$ is convex, closed, and non-empty, then its indicator function ι_C is CCP [4, p.8]. Thus, since $f = \iota_{\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}}$ and $\mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ is a linear subspace of $\mathbb{R}^{q(T_{\text{ini}}+T_{\text{f}})}$ [21], f is CCP [4, p.8]. Similarly, g is convex, closed, since its epigraph is closed, and proper, as g is finite if $w = \text{col}(w_{\text{ini}}, w_{\text{f}})$ for some w_{f} and equals $+\infty$, otherwise.

(ii) Satisfied by design since Φ is positive definite.

(iii) Since $w_{\text{ini}} \in \mathcal{B}|_{[1,T_{\text{ini}}]}$ and $T_{\text{ini}} \geqslant \ell$, any trajectory of the form $w_{\text{ini}} \wedge w_{\text{f}} \in \mathcal{B}|_{[1,T_{\text{ini}}+T_{\text{f}}]}$ is feasible, ensuring $\mathbf{zer}(\partial f + \partial g + \nabla h) \neq \varnothing$.

(iv) This condition is satisfied by design.

By Lemma 2, $w_k \to w^\star \in \mathbf{zer}(\partial f + \partial g + \nabla h)$ as $k \to \infty$. Uniqueness of $w^\star$ follows from Φ being positive definite and the one-to-one correspondence established by Fact 2 between solutions of (6) and (10). ■