



Economic nonlinear model predictive control with reduced complexity: Learning storage and terminal cost functions from data

Shokhjakhon Abdufattokhov^{ID*}, Mario Zanon^{ID}, Alberto Bemporad^{ID}

IMT School for Advanced Studies in Lucca, Piazza S.Francesco 19, Lucca 55100, Lucca, Italy

ARTICLE INFO

Keywords:

Learning-based predictive control
Cost-to-go approximation
Constrained nonlinear systems
Neural networks
Economic MPC

ABSTRACT

Economic nonlinear model predictive control (ENMPC) optimizes general economic objectives for constrained nonlinear systems online. Guaranteeing closed-loop stability typically requires the existence of a storage function satisfying strict dissipativity and appropriate terminal cost and constraints. A long prediction horizon is typically required to achieve good performance, which may result in high computational complexity. This paper proposes a reduced-complexity ENMPC formulation with a prediction horizon of length one and no terminal constraints, in which both the storage function and a Lyapunov terminal cost approximating the long-horizon cost-to-go are learned jointly offline. The resulting controller preserves the asymptotic stability guarantees of the original formulation while significantly reducing the online computational burden. Numerical simulations on two nonlinear benchmarks demonstrate that the proposed scheme achieves near-optimal closed-loop economic performance while reducing considerably the computation time.

1. Introduction

Nonlinear Model Predictive Control (NMPC) has become increasingly popular due to its effectiveness in managing complex dynamic systems with hard constraints on states and manipulated inputs, while delivering good closed-loop performance and providing stability guarantees under appropriate conditions. In its classical form, NMPC is typically formulated as a tracking problem in which the objective is to minimize the deviation from a prescribed steady-state. Economic NMPC (ENMPC) extends this framework by adopting a more general economic stage cost, thereby making the controller particularly attractive for applications involving real-time improvement of operational efficiency [1], energy reduction [2], and profit maximization [3]. Despite such profit-making potential, the implementation of ENMPC faces challenges and limitations with regard to guaranteeing closed-loop stability while maintaining computational tractability.

In a classical NMPC problem, stability is generally achieved by incorporating a Lyapunov-based terminal cost and a suitable terminal constraint set, provided that the stage cost is positive definite around the steady-state Ref. [4]. However, in classical ENMPC, the economic stage cost is generally not positive definite with respect to the economically optimal steady-state. As a consequence, the optimal value function does not satisfy a Lyapunov decrease property and stability certificates rely on strict dissipativity [5–7]. If the system satisfies strict dissipativity for some storage function, it is possible to design rotated

versions of the stage and terminal costs that are positive definite with respect to the steady-state. This leads to a rotated ENMPC (RENMPC) formulation that produces the same optimal solutions as the original ENMPC scheme while benefiting from: (i) well-established Lyapunov results that guarantee that the closed-loop system is asymptotically stable [8,9]; and (ii) efficient numerical algorithms to solve the resulting optimization problem in real time [10–12]. However, (i) certifying dissipativity requires complex storage functions [13] and is often intractable; and (ii) the resulting RENMPC may become computationally intensive for long prediction horizons.

To reduce complexity, one viable option is to approximate the optimal value function by a Lyapunov terminal cost, thereby obtaining a fast, yet stabilizing, RENMPC formulation that accurately approximates the cost-to-go beyond the desired short prediction horizon [14]. A further complexity reduction can be achieved by removing the terminal constraint, at the expense of reducing the size of the certifiable region of attraction [15]. Unfortunately, deriving an analytical and non-conservative Lyapunov terminal cost is generally a challenging task [16]. To mitigate this challenge, several data-driven approaches have been proposed to learn Lyapunov functions directly from data [17–20]. Recent studies [21,22] explore the verification of dissipativity properties directly from system data. Other works [23,24] investigate learning terminal cost functions to accurately estimate the long-term cost-to-go.

* Corresponding author.

E-mail addresses: s.abdulfattokhov@imtlucca.it (S. Abdulfattokhov), mario.zanon@imtlucca.it (M. Zanon), alberto.bemporad@imtlucca.it (A. Bemporad).

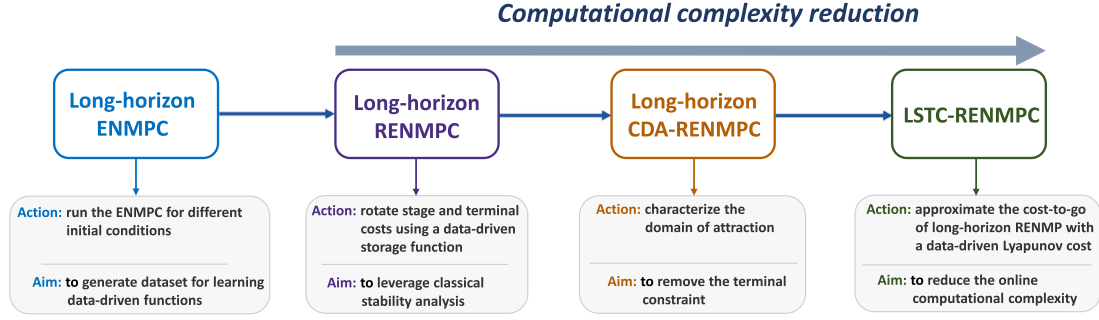


Fig. 1. Schematic workflow for constructing LSTC-RENMP from Classical ENMPC.

In this paper, we present a learning-based methodology for designing an approximate, yet computationally light, RENMPC scheme with a prediction horizon of length one without using terminal constraints. We extend our recent work [25], which considered learning a terminal cost for horizon-one NMPC with tracking costs, to economic objectives and to the learning of a storage function. We propose a two-step approximation scheme: first, the terminal constraint is removed by characterizing the domain of attraction, thereby yielding RENMPC with *Characterized Domain of Attraction* (CDA-RENMPC) while preserving asymptotic closed-loop stability; second, the long-horizon cost-to-go of CDA-RENMPC is approximated by a Lyapunov terminal cost function to obtain a one-horizon controller.

We employ feedforward neural networks to construct both the terminal cost and the storage function, motivated by their universal approximation capability [26] and their increasingly recognized role in reducing the online computational burden of MPC [27]. Specifically, the terminal cost is parameterized as a quadratic function of the terminal state, with positive definiteness enforced by construction. The entries of the Hessian matrix of this function are generated by a feedforward neural network, while a separate network is used to represent the storage function. Contrary to [21,24], which learn the storage function and the terminal cost through reinforcement learning, respectively, we train both networks jointly offline through supervised learning embedded in a constrained nonlinear optimization problem. The Lyapunov and dissipativity conditions are enforced simultaneously on a common finite dataset generated from the closed-loop solutions of the classical long-horizon ENMPC scheme. The characterized domain of attraction is subsequently constructed from the training dataset by retaining the sampled initial states that satisfy the corresponding CDA inclusion condition. Accordingly, the stability guarantees established in this work apply to the finite set of data points that constitute the resulting sample-based characterized domain of attraction.

We refer to the proposed approach as RENMPC with *learned storage and terminal cost* functions (LSTC-RENMPC). The overall step-by-step progression from the classical long-horizon ENMPC formulation to the one-horizon LSTC-RENMPC scheme is illustrated in Fig. 1. We perform numerical simulations on nonlinear benchmark control problems and show that the proposed control scheme attains significant reductions in computational complexity compared to the long-horizon CDA-RENMPC formulation, while achieving comparable closed-loop performance.

The remainder of this paper is structured as follows. Section 2 introduces the key ingredients to obtain the RENMPC problem with a prediction horizon of length one. Section 3 is devoted to learning a storage function that satisfies the dissipativity inequality and a Lyapunov-based approximation of the cost-to-go function. Section 4 demonstrates the effectiveness of the proposed approach through numerical simulations, and concluding remarks are provided in Section 5.

2. Preliminaries and problem formulation

This section introduces the mathematical setting and preliminaries used throughout the paper. We first present the classical ENMPC problem formulation, followed by the dissipativity-based cost-rotation framework that leads to an equivalent rotated formulation. Finally, we introduce the ingredients that will allow us to develop a reduced-complexity controller.

In our formulations, we consider the following discrete-time nonlinear system of the form

$$x_{t+1} = f(x_t, u_t) \quad (1)$$

where $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$, and $x_t \in \mathbb{R}^{n_x}$, $u_t \in \mathbb{R}^{n_u}$ denote the state and input at a time instant t , respectively. The states and inputs of the system are constrained within compact sets $\mathcal{X}$ and $\mathcal{U}$, respectively.

2.1. Economic nonlinear MPC

A classical ENMPC problem with a prediction horizon N and terminal components is formulated as

$$\begin{aligned} J_N^*(p_t) = \min_{U_t, X_t} \quad & J_N(U_t, X_t, p_t) \\ \text{s.t.} \quad & x_{0|t} = \mathcal{M}_x p_t \\ & x_{k+1|t} = f(x_{k|t}, u_{k|t}) \quad k \in \mathbb{I}_0^{N-1} \\ & u_{k|t} \in \mathcal{U} \quad k \in \mathbb{I}_0^{N-1} \\ & x_{k|t} \in \mathcal{X} \quad k \in \mathbb{I}_1^{N-1} \\ & x_{N|t} \in \mathcal{X}_T(p_t) \end{aligned} \quad (2)$$

where the objective is given by

$$J_N(U_t, X_t, p_t) = \sum_{k=0}^{N-1} \ell(x_{k|t}, u_{k|t}, p_t) + \ell_T(x_{N|t}, p_t),$$

$p_t = [x_t' \ \theta_t']' \in \mathbb{R}^{n_p}$ is a vector belonging to a compact set $\mathcal{P}$ that includes the current state x_t and another vector θ_t of parameters defining the economic cost and terminal state constraint, $\mathcal{M}_x$ is a real matrix used to define the initial state, $U_t = (u_{0|t}, \dots, u_{N-1|t})$ is the sequence of manipulated variables $u_{k|t} \in \mathbb{R}^{n_u}$, $X_t = (x_{0|t}, \dots, x_{N|t})$ is the sequence of predicted states $x_{k|t} \in \mathbb{R}^{n_x}$ evaluated from the initial state $x_{0|t}$, $\mathbb{I}_a^b$ is the set of all integers in the interval $[a, b]$, $\ell : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}$ is a stage cost function that measures the economic performance of the system, $\ell_T : \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}$ is a terminal cost function that penalizes the terminal state $x_{N|t}$ on the compact terminal set $\mathcal{X}_T(p_t) \subseteq \mathcal{X}$. Such a set determines a neighborhood of the optimal steady state $x_s(p_t)$, which is defined as follows:

$$x_s(p_t), u_s(p_t) = \arg \min_{x, u} \ell(x, u, p_t) \quad (3)$$

$$\begin{aligned} \text{s. t. } & x = f(x, u) \\ & x \in \mathcal{X} \\ & u \in \mathcal{U} \end{aligned}$$

where the optimal solution pair $(x_s(p_t), u_s(p_t))$ is assumed to be unique for each $p_t \in \mathcal{P}$.

In the sequel, $\mathcal{F}_N(p_t)$ denotes the set of initial states for which (2) is feasible, and we write $U_t^* = (u_{0|t}^*, \dots, u_{N-1|t}^*)$ and $X_t^* = (x_{0|t}^*, \dots, x_{N|t}^*)$ for the corresponding optimal input and state sequences. The ENMPC feedback law is defined in a receding-horizon fashion by applying the first optimal control action $u_t = u_{0|t}^*$ to the controlled system (1).

We introduce the next two assumptions that will make it possible to prove the stability of the closed-loop system under the control law defined by (2).

Assumption 1. The functions f , ℓ and ℓ_T are twice continuously differentiable on $\mathcal{X} \times \mathcal{U}$, $\mathcal{X} \times \mathcal{U} \times \mathcal{P}$, and $\mathcal{X}_T(p_t) \times \mathcal{P}$, respectively.

Assumption 2. There exists a function $u_T : \mathcal{X}_T(p_t) \times \mathcal{P} \rightarrow \mathcal{U}$ satisfying

$$\begin{aligned} \ell_T(f(x, u_T(x, p_t)), p_t) - \ell_T(x, p_t) &\leq -\ell(x, u_T(x, p_t), p_t) \\ &+ \ell(x_s(p_t), u_s(p_t), p_t) \end{aligned} \quad (4)$$

$\forall x \in \mathcal{X}_T(p_t)$ and $\forall p \in \mathcal{P}$.

We recall next the standard notion of class- $\mathcal{K}$ functions.

Definition 1. A function $\mu : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is a class- $\mathcal{K}$ function if it is continuous, strictly increasing, and zero at the origin.

Under Assumptions 1 and 2, constant parameters Θ_t , and the additional assumption that, for all $p_t \in \mathcal{P}$ we have $\ell(x_s(p_t), u_s(p_t)) = 0$ and $\ell(x, u) \geq \alpha(\|x - x_s(p_t)\|)$, with $\alpha : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$, $\alpha \in \mathcal{K}$, standard tracking (i.e., non-economic) NMPC results [4] prove that the optimal value function $J_N^*(p_t)$ is a Lyapunov function on $\mathcal{F}_N(p_t)$. Consequently, the optimal feedback law defined by (2) asymptotically stabilizes the system (1). However, in ENMPC the stage cost $\ell(x, u, p_t)$ is not positive definite. As a consequence, the classical Lyapunov arguments used in tracking NMPC cannot be directly applied to establish closed-loop stability. To recover Lyapunov-based stability arguments in the economic setting, dissipativity-based analysis is commonly employed with respect to a prescribed supply rate. In this framework, a storage function is used to rotate the economic cost, leading to a rotated-cost formulation whose optimal value function can be used as a Lyapunov function under appropriate assumptions. The following definition recalls the notion of strict dissipativity.

Definition 2. Let $s : \mathcal{X} \times \mathcal{U} \times \mathcal{P} \rightarrow \mathbb{R}$ be a given supply rate. The system (1) is strictly dissipative with respect to s if there exist a storage function $\lambda : \mathcal{X} \times \mathcal{P} \rightarrow \mathbb{R}$ and a positive definite function $\rho : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ such that

$$\lambda(f(x, u), p_t) - \lambda(x, p_t) \leq -\rho(\|x - x_s(p_t)\|) + s(x, u, p_t), \quad (5)$$

for all $x \in \mathcal{X}$, $u \in \mathcal{U}$, and $p_t \in \mathcal{P}$.

Assumption 3. The storage function $\lambda(x, p_t)$ is continuous on $\mathcal{X} \times \mathcal{P}$.

In the ENMPC formulation considered in this paper, the prescribed supply rate is induced by the economic stage cost relative to the economically optimal steady state.

Assumption 4. System (1) is strictly dissipative with respect to the supply rate

$$s(x, u, p_t) := \ell(x, u, p_t) - \ell(x_s(p_t), u_s(p_t), p_t), \quad (6)$$

for all $x \in \mathcal{X}$, $u \in \mathcal{U}$, and $p_t \in \mathcal{P}$.

2.2. Rotated economic nonlinear MPC

To leverage dissipativity in the stability analysis of classical ENMPC, the finite-horizon objective is reformulated by incorporating the storage function λ into the cost. In particular, we introduce the *rotated* stage and terminal cost functions

$$\begin{aligned} \mathcal{L}(x, u, p_t) &:= \ell(x, u, p_t) - \ell(x_s(p_t), u_s(p_t), p_t) \\ &+ \lambda(x, p_t) - \lambda(f(x, u), p_t) \end{aligned} \quad (7)$$

$$\begin{aligned} \mathcal{L}_T(x, p_t) &:= \ell_T(x, p_t) + \lambda(x, p_t) - \ell_T(x_s(p_t), p_t) \\ &- \lambda(x_s(p_t), p_t) \end{aligned} \quad (8)$$

which then leads to the corresponding rotated version of the classical ENMPC problem (2), referred to as classical RENMPC problem in the sequel, defined as

$$\begin{aligned} \bar{J}_N^*(p_t) &= \min_{U_t, X_t} \bar{J}_N(U_t, X_t, p_t) \\ \text{s. t. } & x_{0|t} = \mathcal{M}_x p_t \\ & x_{k+1|t} = f(x_{k|t}, u_{k|t}) \quad k \in \mathbb{I}_0^{N-1} \\ & u_{k|t} \in \mathcal{U} \quad k \in \mathbb{I}_0^{N-1} \\ & x_{k|t} \in \mathcal{X} \quad k \in \mathbb{I}_1^{N-1} \\ & x_{N|t} \in \mathcal{X}_T(p_t) \end{aligned} \quad (9)$$

where the only difference with respect to (2) is in the cost

$$\bar{J}_N(U_t, X_t, p_t) = \sum_{k=0}^{N-1} \mathcal{L}(x_{k|t}, u_{k|t}, p_t) + \mathcal{L}_T(x_{N|t}, p_t).$$

Before using the rotated costs in later developments, we recall in the following theorem that the rotation does not change the primal solution. This fact is important because it guarantees that any controller derived from the rotated problem implements the same closed-loop input sequence as the original ENMPC, while enabling a Lyapunov-based stability analysis.

Theorem 1 (See [9]). *If Assumptions 1–4 hold, then the primal solutions of ENMPC (2) and RENMPC (9) laws are identical.*

Proof. The rotated objective cost reads

$$\begin{aligned} \bar{J}_N(U_t, X_t, p_t) &:= \sum_{k=0}^{N-1} \mathcal{L}(x_{k|t}, u_{k|t}, p_t) + \mathcal{L}_T(x_{N|t}, p_t) \\ &= \sum_{k=0}^{N-1} (\ell(x_{k|t}, u_{k|t}, p_t) - \ell(x_s(p_t), u_s(p_t), p_t) \\ &\quad + x_s \lambda(x_{k|t}, p_t) - \lambda(x_{k+1|t}, p_t)) \\ &\quad + x_s \ell_T(x_{N|t}, p_t) - \ell_T(x_s(p_t), p_t) \\ &\quad + x_s \lambda(x_{N|t}, p_t) - \lambda(x_s(p_t), p_t) \\ &= \left(\sum_{k=0}^{N-1} \ell(x_{k|t}, u_{k|t}, p_t) + \ell_T(x_{N|t}, p_t) \right) \\ &\quad + x_s \lambda(x_{0|t}, p_t) - N \ell(x_s(p_t), u_s(p_t), p_t) \\ &\quad - x_s \ell_T(x_s(p_t), p_t) - \lambda(x_s(p_t), p_t) \\ &= J_N(U_t, X_t, p_t) + \lambda(x_{0|t}, p_t) \\ &\quad - x_s N \ell(x_s(p_t), u_s(p_t), p_t) - \ell_T(x_s(p_t), p_t) \\ &\quad - x_s \lambda(x_s(p_t), p_t). \end{aligned}$$

This proves that objective functions $\bar{J}_N(U_t, X_t, p_t)$ and $J_N(U_t, X_t, p_t)$ differ from each other only by constant terms. Consequently, the problems in (2) and (9) have the same primal solution.

2.3. RENMPC with characterized domain of attraction

The main difficulty associated with RENMPC (9) is the construction of suitable terminal ingredients. In particular, determining an

appropriate terminal region is often nontrivial, and explicitly enforcing the terminal constraint may require longer prediction horizons, both of which limit computational efficiency in real-time applications. An approach that allows one to remove the terminal constraints has been proposed in [15] for tracking NMPC. The authors of [25] extended the idea proposed in [15] to the case of a parameter-dependent formulation. This ensures implicit satisfaction of the terminal constraint and yields stability of the problem without the terminal constraint by characterizing a positively invariant subset of the region of attraction. Following the procedure proposed in [25] for tracking NMPC, we define the following formulation

$$\begin{aligned} \bar{J}_{CDA}^*(p_t) = \min_{U_t, \mathcal{X}_t} \sum_{k=0}^{N-1} \mathcal{L}(x_{k|t}, u_{k|t}, p_t) + \mathcal{L}_T(x_{N|t}, p_t) \quad (10) \\ \text{s. t. } x_{0|t} = \mathcal{M}_x p_t \\ x_{k+1|t} = f(x_{k|t}, u_{k|t}) \quad k \in \mathbb{I}_0^{N-1} \\ u_{k|t} \in \mathcal{U} \quad k \in \mathbb{I}_0^{N-1} \\ x_{k|t} \in \mathcal{X} \quad k \in \mathbb{I}_1^N \end{aligned}$$

and we call it RENMPC with *Characterized Domain of Attraction* (CDA-RENMPC) in the sequel. Since this formulation serves as the stability-oriented reference problem for the proposed low-complexity controller, we denote its optimal cost by $\bar{J}_{CDA}^*$ to distinguish it from that of (9), although $\bar{J}_{CDA}^*$ and $\bar{J}_N^*$ coincide on the characterized domain of attraction. Unlike the tracking NMPC setting in [25], where the characterized-domain formulation is used within the data-generation procedure, CDA-RENMPC is used here as the terminal-constraint-free reference problem for learning the terminal ingredients and evaluating the proposed controller.

Next, we introduce the two technical ingredients establishing basic lower- and upper-bound properties of the rotated stage and terminal cost functions, which together are instrumental for proving the asymptotic stability of the CDA-RENMPC scheme.

Lemma 2 (Extension of [9, Lemma 10]). *Let Assumptions 1,3,4 hold, then the rotated stage cost function $\mathcal{L}(x, u, p_t)$ satisfies*

$$\mathcal{L}(x, u, p_t) \geq \mu_1(\|x - x_s(p_t)\|) \quad (11)$$

for all $(x, u, p_t) \in \mathcal{X} \times \mathcal{U} \times \mathcal{P}$, with $\mu_1 \in \mathcal{K}$.

Proof. By rearranging the terms in (5) we obtain

$$\mathcal{L}(x, u, p_t) \geq \rho(\|x - x_s(p_t)\|), \quad \forall (x, u, p_t) \in \mathcal{X} \times \mathcal{U} \times \mathcal{P}. \quad (12)$$

Moreover [9, Lemma 10], there exists $\mu_1 \in \mathcal{K}$ such that

$$\rho(\|x - x_s(p_t)\|) \geq \mu_1(\|x - x_s(p_t)\|), \quad \forall (x, p_t) \in \mathcal{X} \times \mathcal{P}. \quad (13)$$

Lemma 3 (Extension of [9] Results). *This lemma combines and extends the arguments of [9, Lemmas 9 and 13]. Let Assumptions 1–4 hold and let the terminal set $\mathcal{X}_T(p_t)$ be chosen within a sublevel set of the rotated terminal cost, i.e.,*

$$\mathcal{X}_T(p_t) \subseteq \{x \in \mathcal{X} : \mathcal{L}_T(x, p_t) \leq \alpha(p_t)\}, \quad \forall p_t \in \mathcal{P}, \quad (14)$$

where $\alpha : \mathcal{P} \rightarrow \mathbb{R}_{>0}$ specifies the corresponding level. Then the set $\mathcal{X}_T(p_t)$ is forward invariant under the terminal control law $u_T(x, p_t)$, and the function $\mathcal{L}_T(x, p_t)$ is a Lyapunov function on $\mathcal{X}_T(p_t)$, i.e., for all $x \in \mathcal{X}_T(p_t)$ and all $p_t \in \mathcal{P}$,

$$f(x, u_T(x, p_t)) \in \mathcal{X}_T(p_t), \quad (15a)$$

$$\mathcal{L}_T(f(x, u_T(x, p_t)), p_t) - \mathcal{L}_T(x, p_t) \leq -\mathcal{L}(x, u_T(x, p_t), p_t), \quad (15b)$$

$$\mu_2(\|x - x_s(p_t)\|) \leq \mathcal{L}_T(x, p_t) \leq \mu_3(\|x - x_s(p_t)\|), \quad (15c)$$

where μ_2 and μ_3 are class $\mathcal{K}$ functions.

Proof. Adding the term $\lambda(f(x, u_T(x, p_t)), p_t) - \lambda(x, p_t)$ to both sides of (4), and adding and subtracting $\mathcal{L}_T(x_s(p_t), p_t) + \lambda(x_s(p_t), p_t)$, yields (15b) using the definitions (7) and (8). Moreover, by Definition 2 and Assumption 4, the rotated stage cost satisfies $\mathcal{L}(x, u, p_t) \geq \rho(\|x - x_s(p_t)\|) \geq 0$ for all $x \in \mathcal{X}$, $u \in \mathcal{U}$, and $p_t \in \mathcal{P}$. Hence, (15b) implies that $\mathcal{L}_T(f(x, u_T(x, p_t)), p_t) \leq \mathcal{L}_T(x, p_t)$ for all $x \in \mathcal{X}_T(p_t)$. In view of (14), this yields $f(x, u_T(x, p_t)) \in \mathcal{X}_T(p_t)$ for all $x \in \mathcal{X}_T(p_t)$, thereby proving forward invariance of $\mathcal{X}_T(p_t)$ under $u_T(x, p_t)$. Finally, $\mathcal{L}_T(x, p_t)$ is continuous on the compact set $\mathcal{X}_T(p_t)$ for all $p_t \in \mathcal{P}$, and $\mathcal{L}_T(x_s(p_t), p_t) = 0$ by (8). Therefore, there exist class $\mathcal{K}$ functions μ_2 and μ_3 such that (15c) holds [9, Lemma 13]. Consequently, (15a), (15b), and (15c) establish that $\mathcal{L}_T(x, p_t)$ is a Lyapunov function on $\mathcal{X}_T(p_t)$ under $u_T(x, p_t)$.

Theorem 4 ([25, Theorem 1]).

Suppose Assumptions 1–4 hold for every constant parameter vector Θ_t under consideration. Then the CDA-RENMPC controller defined by (10) asymptotically stabilizes system (1) at the economically optimal steady state $x_s(p_t)$ for every initial state $x_{0|t} \in \bar{\Omega}_N(p_t) \subseteq \mathcal{F}_N(p_t)$, where

$$\bar{\Omega}_N(p_t) := \{x_{0|t} \in \mathcal{X} : \bar{J}_{CDA}^*(p_t) \leq \mathcal{L}(x_{0|t}, u_{0|t}^*, p_t) + C_N(p_t)\}, \quad (16)$$

with

$$C_N(p_t) := (N-1)d(p_t) + \alpha(p_t), \quad (17)$$

where $d(p_t) > 0$ is defined as

$$d(p_t) := \inf_{\substack{x \in \mathcal{X} \setminus \mathcal{X}_T(p_t) \\ u \in \mathcal{U}}} \mathcal{L}(x, u, p_t), \quad (18)$$

and $\alpha(p_t)$ satisfies (14).

The proof is omitted for brevity, as it follows directly from the arguments in [25, Theorem 1] together with Lemmas 2 and 3.

2.4. Complexity reduction

In case the required prediction horizon N is long, solving the CDA-RENMPC problem (10) can be computationally demanding. This motivates attempting to formulate a problem with a shorter prediction horizon and a properly defined terminal cost that still stabilizes the system and does not deteriorate closed-loop performance significantly. To this end, we take advantage of Bellman's principle of optimality and reformulate the CDA-RENMPC problem into the following equivalent problem with prediction horizon $N = 1$

$$\begin{aligned} \bar{J}_{CDA}^*(p_t) = \min_{u_{0|t}, x_{0|t}, x_{1|t}} \mathcal{L}(x_{0|t}, u_{0|t}, p_t) + \mathcal{V}(x_{1|t}, p_t) \quad (19) \\ \text{s. t. } x_{0|t} = \mathcal{M}_x p_t \\ x_{1|t} = f(x_{0|t}, u_{0|t}) \\ u_{0|t} \in \mathcal{U} \\ x_{1|t} \in \mathcal{X} \end{aligned}$$

where $\mathcal{V} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}$ is the cost-to-go defined as

$$\begin{aligned} \mathcal{V}(x_{1|t}, p_t) = \min_{U_{1|t}, \mathcal{X}_{1|t}} \sum_{k=1}^{N-1} \mathcal{L}(x_{k|t}, u_{k|t}, p_t) + \mathcal{L}_T(x_{N|t}, p_t) \quad (20) \\ \text{s. t. } x_{k+1|t} = f(x_{k|t}, u_{k|t}) \quad k \in \mathbb{I}_1^{N-1} \\ u_{k|t} \in \mathcal{U} \quad k \in \mathbb{I}_1^{N-1} \\ x_{k|t} \in \mathcal{X} \quad k \in \mathbb{I}_2^N \end{aligned}$$

with $U_{1|t} = (u_{1|t}, \dots, u_{N-1|t})$ and $X_{1|t} = (x_{1|t}, \dots, x_{N|t})$.

If the cost-to-go function $\mathcal{V}$ were available in simple explicit form, the CDA-RENMPC problem with horizon $N = 1$ defined in (19) could be solved substantially faster than the long-horizon CDA-RENMPC formulation (10). However, in practice, obtaining explicit expressions for the storage function λ and the cost-to-go function $\mathcal{V}$ is generally intractable. Therefore, we adopt a learning-based approach to construct data-driven surrogates from closed-loop data, as detailed below.

3. RENMPC with learned storage and terminal cost functions

This section proposes a learning-based construction of the CDA-RENMPC scheme (19), that we will refer to as Learned-Storage and Terminal-Cost RENMPC (LSTC-RENMPC). Specifically, we show how the storage function and the long-horizon cost-to-go can be approximated from closed-loop data using neural networks, thereby yielding data-driven versions of the rotated stage and terminal cost functions $\mathcal{L}$ and $\mathcal{L}_T$, as well as a simple approximation of the cost-to-go function $\mathcal{V}$ in (20). The parameters of the two networks are trained jointly offline in a supervised manner, in a way that enforces the associated dissipativity and Lyapunov conditions on a common dataset generated from closed-loop simulations of the classical long-horizon ENMPC. These learned components are then used to obtain the LSTC-RENMPC controller, which preserves the key properties of the long-horizon rotated formulation while substantially reducing the online computational burden.

3.1. Storage function parameterization

Since an explicit expression for the storage function $\lambda(x, p_t)$ is not available, we replace $\lambda(x, p_t)$ with a surrogate $\hat{\lambda}(x, p_t; \theta)$ to be learned. We use $\hat{\lambda}$ to define surrogates $\hat{\mathcal{L}}$ and $\hat{\mathcal{L}}_T$ of the rotated stage and terminal costs to be learned in a data-driven fashion. The rotated costs to be learned are defined as

$$\begin{aligned} \hat{\mathcal{L}}(x, u, p_t; \theta) &:= s(x, u, p_t) + \hat{\lambda}(x, p_t; \theta) \\ &\quad - \hat{\lambda}(f(x, u), p_t; \theta), \end{aligned} \quad (21)$$

$$\begin{aligned} \hat{\mathcal{L}}_T(x, p_t; \theta) &:= \mathcal{L}_T(x, p_t) + \hat{\lambda}(x, p_t; \theta) \\ &\quad - \mathcal{L}_T(x_s(p_t), p_t) - \hat{\lambda}(x_s(p_t), p_t; \theta), \end{aligned} \quad (22)$$

where $\hat{\lambda}$ is defined as a feedforward neural network, i.e., $\hat{\lambda}(x, p_t; \theta) := \Psi(\xi_t; \theta, \mathcal{H}, \mathcal{A}, \eta)$ with the network input $\xi_t := [x^\top, p_t^\top]^\top$. The network is defined as

$$\begin{aligned} \mathcal{G}_{\theta_h}(z_{h-1}) &:= w_h z_{h-1} + b_h, \\ z_h &:= \mathcal{A}_h(\mathcal{G}_{\theta_h}(z_{h-1})), \quad h \in \mathbb{I}_1^{H+1}, \end{aligned}$$

with $z_0 = \xi_t$, where $\theta := [\theta_1, \theta_2, \dots, \theta_{H+1}]$, $\mathcal{A} := [\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_{H+1}]$, $\eta := [\eta_1, \eta_2, \dots, \eta_{H+1}]$, and $\mathcal{H}$ denotes the number of hidden layers, η_h is the number of neurons in layer h , $\mathcal{A}_h$ is the activation function, and each layer map $\mathcal{G}_{\theta_h}$ is affine with parameters $\theta_h = [w_h, b_h]$. More comprehensive details on neural network architectures, including the choice of activation functions, training methodology, and implementation considerations, can be found in [28].

3.2. Lyapunov cost-to-go parameterization

Building on the parameterization of the storage function, we now construct a tractable approximation of the value term that augments the one-horizon CDA-RENMPC reformulation. To clarify the learning target, we exploit the equivalence between the classical ENMPC and its rotated counterpart established in Theorem 1 and rewrite the long-horizon cost-to-go in a form that separates the storage contribution from the remaining rotated-performance component. In particular, from (20) it follows that

$$\begin{aligned} \mathcal{V}(x_{1|t}^*, p_t) &= \sum_{k=1}^{N-1} \mathcal{L}(x_{k|t}^*, u_{k|t}^*, p_t) + \mathcal{L}_T(x_{N|t}^*, p_t) \\ &= \sum_{k=1}^{N-1} \left(\mathcal{L}(x_{k|t}^*, u_{k|t}^*, p_t) - \mathcal{L}(x_s(p_t), u_s(p_t), p_t) \right. \\ &\quad \left. + x_s \mathcal{L}(x_{k|t}^*, p_t) - \lambda(x_{k+1|t}^*, p_t) \right) + \mathcal{L}_T(x_{N|t}^*, p_t) \\ &\quad - x_s \mathcal{L}_T(x_s(p_t), p_t) + \lambda(x_{N|t}^*, p_t) - \lambda(x_s(p_t), p_t) \\ &= \sum_{k=1}^{N-1} \left(\mathcal{L}(x_{k|t}^*, u_{k|t}^*, p_t) - \mathcal{L}(x_s(p_t), u_s(p_t), p_t) \right) \end{aligned}$$

$$\begin{aligned} &+ x_s \mathcal{L}_T(x_{N|t}^*, p_t) - \mathcal{L}_T(x_s(p_t), p_t) \\ &+ x_s \lambda(x_{1|t}^*, p_t) - \lambda(x_s(p_t), p_t) \\ &= \mathcal{V}(x_{1|t}^*, p_t) + \lambda(x_{1|t}^*, p_t) - \lambda(x_s(p_t), p_t) \end{aligned}$$

where $\mathcal{V}(x_{1|t}^*, p_t)$ denotes the cost-to-go of the original problem (before rotation), and is defined as

$$\begin{aligned} \mathcal{V}(x_{1|t}^*, p_t) &:= \sum_{k=1}^{N-1} \left(\mathcal{L}(x_{k|t}^*, u_{k|t}^*, p_t) - \mathcal{L}(x_s(p_t), u_s(p_t), p_t) \right) \\ &\quad + x_s \mathcal{L}_T(x_{N|t}^*, p_t) - \mathcal{L}_T(x_s(p_t), p_t). \end{aligned} \quad (23)$$

We approximate the cost to go by a quadratic Lyapunov candidate, since this class yields a simple representation that captures the local curvature of the value function around the optimal steady state with positive definiteness enforced by construction:

$$\hat{\mathcal{V}}(x, p_t; \theta) := (x - x_s(p_t))^\top \hat{P}(p_t; \theta) (x - x_s(p_t)), \quad (24)$$

$$\hat{P}(p_t; \theta) := \hat{L}(p_t; \theta)^\top (p_t; \theta) + \epsilon I, \quad (25)$$

where $\epsilon > 0$ is a small regularization constant and I denotes the identity matrix. The factorization in (25) guarantees $\hat{P}(p_t; \theta) > 0$ for all p , ensuring that $\hat{\mathcal{V}}(x, p_t; \theta)$ is positive definite with respect to $x = x_s(p_t)$ by construction. The nonzero entries of the matrix $\hat{L}(p_t; \theta)$ are generated by a feedforward neural network $\Phi(p_t; \theta, \mathcal{H}, \mathcal{A}, \eta)$.

3.3. Learning problem

To learn the network parameters θ and θ from closed-loop trajectories of the classical long-horizon ENMPC, we collect a dataset

$$\mathbb{D} = \{p^i, u_0^i, x_1^i, p^i, s_0^i, V_1^i\}_{i=1}^M$$

where p^i contains x_0^i , by solving (2) for each initial condition p^i and

$$\begin{aligned} \rho^i &= (x_0^i - x_s(p^i))^\top \epsilon I (x_0^i - x_s(p^i)), \\ s_0^i &= \mathcal{L}(x_0^i, u_0^i, p^i) - \mathcal{L}(x_s(p^i), u_s(p^i), p^i), \\ V_1^i &= \mathcal{V}(x_1^i, p^i). \end{aligned}$$

We then solve the following constrained supervised learning problem

$$\begin{aligned} \min_{\theta, \theta} \quad & \gamma_1 \|\theta\|_2^2 + \gamma_2 \|\theta\|_2^2 + \frac{1}{M} \sum_{i=1}^M \phi(V_1^i, \hat{\mathcal{V}}(x_1^i, p^i; \theta) \\ & - x_s \hat{\lambda}(x_1^i, p^i; \theta) + \hat{\lambda}(x_s(p^i), p^i; \theta)) \end{aligned} \quad (26a)$$

$$\text{s. t.} \quad \hat{\lambda}(x_1^i, p^i; \theta) - \hat{\lambda}(x_s(p^i), p^i; \theta) \leq -\rho^i + s_0^i \quad (26b)$$

$$\hat{\mathcal{V}}(x_1^i, p^i; \theta) \leq \hat{C}_N(p^i; \theta) \quad (26c)$$

$$\hat{\mathcal{V}}(x_1^i, p^i; \theta) - \hat{\mathcal{V}}(x_0^i, p^i; \theta) \leq -\hat{L}(x_0^i, u_0^i, p^i; \theta). \quad (26d)$$

where the loss term ϕ penalizes deviations between V_1^i and $\hat{\mathcal{V}}(x_1^i, p^i; \theta) - \hat{\lambda}(x_1^i, p^i; \theta) + \hat{\lambda}(x_s(p^i), p^i; \theta)$, the quadratic penalties $\gamma_1 \|\theta\|_2^2$ and $\gamma_2 \|\theta\|_2^2$ regularize the network parameters, and the three constraints enforce, respectively, a dissipativity inequality, a level-set bound, and a decrease condition for the learned Lyapunov candidate. The bound in (26c) uses

$$\hat{C}_N(p^i; \theta) := (N-1)\hat{d}(p^i; \theta) + \hat{\alpha}(p^i; \theta), \quad (27)$$

with

$$\hat{d}(p^i; \theta) := \inf_{x \in \mathcal{X} \setminus \mathcal{X}_T(p^i), u \in \mathcal{U}} \hat{\mathcal{L}}(x, u, p^i; \theta), \quad (28)$$

$$\hat{\alpha}(p^i; \theta) := \max_{x \in \mathcal{X}_T(p^i)} \hat{\mathcal{L}}_T(x, p^i; \theta). \quad (29)$$

In order to implement a standard gradient-based training procedure, we replace the constrained problem with a penalty formulation in which constraint violations are added to the loss:

$$\theta^*, \theta^* := \arg \min_{\theta, \theta} \gamma_1 \|\theta\|_2^2 + \gamma_2 \|\theta\|_2^2$$

$$\begin{aligned}
& + \frac{1}{M} \sum_{i=1}^M \phi \left(V_1^i, \hat{V}(x_1^i, p^i; \theta) - \hat{\lambda}(x_1^i, p^i; \theta) \right. \\
& \left. + \hat{\lambda}(x_s(p^i), p^i; \theta) \right) + \kappa_1 \mathcal{E}_1^i(\theta) \\
& + \kappa_2 \mathcal{E}_2^i(\theta) + \kappa_3 \mathcal{E}_3^i(\theta)
\end{aligned} \quad (30)$$

where

$$\begin{aligned}
\mathcal{E}_1^i(\theta) &:= \max \{ \hat{\lambda}(x_1^i, p^i; \theta) - \hat{\lambda}(x_0^i, p^i; \theta) + \rho^i - s_0^i, 0 \}, \\
\mathcal{E}_2^i(\theta, \theta) &:= \max \{ \hat{V}(x_1^i, p^i; \theta) - \hat{C}_N(p^i; \theta), 0 \}, \\
\mathcal{E}_3^i(\theta, \theta) &:= \max \{ \hat{V}(x_1^i, p^i; \theta) - \hat{V}(x_0^i, u_0^i, p^i; \theta), 0 \}
\end{aligned}$$

and $\kappa_1, \kappa_2, \kappa_3 > 0$ are the penalty parameters of the imposed constraints. Since feasibility of (26) cannot be guaranteed for arbitrary datasets and network parameterizations, Algorithm 1 repeatedly solves (30) until the three violation terms vanish on both the training and test datasets. This criterion prioritizes satisfaction of the imposed conditions on the available samples, possibly at the expense of exact reproduction of the long-horizon cost-to-go, while preserving the Lyapunov behavior of $\hat{V}$ used to establish stability of the LSTC-RENMPC law.

Algorithm 1: Learning $\hat{\lambda}$ and $\hat{V}$

Input: $\mathbb{D}_{\text{train}}$ and $\mathbb{D}_{\text{test}}$
Output: $\hat{\lambda}(x, p; \theta^*)$ and $\hat{V}(x, p; \theta^*)$
1 **Initialization:** $M_{\text{train}} = \text{length}(\mathbb{D}_{\text{train}})$ and $M_{\text{test}} = \text{length}(\mathbb{D}_{\text{test}})$;
2 **repeat**
3 specify architectures of $\Psi(\xi_i; \theta, \mathcal{H}, \mathcal{A}, \eta)$ and $\Phi(p_i; \theta, \mathcal{H}, \mathcal{A}, \eta)$;
4 initialize parameters θ and θ ;
5 set regularizing parameters $\epsilon > 0$, $\gamma_1 > 0$, $\gamma_2 > 0$, and
6 penalizing parameters $\kappa_1 > 0$, $\kappa_2 > 0$ and $\kappa_3 > 0$;
7 solve the problem (30) for $M = M_{\text{train}}$ to obtain θ^* and θ^* ;
8 **until** $\sum_{i=1}^{M_{\text{train}}} \mathcal{E}_1^i(\theta^*) + \mathcal{E}_2^i(\theta^*, \theta^*) + \mathcal{E}_3^i(\theta^*, \theta^*) = 0$ and
 $\sum_{i=1}^{M_{\text{test}}} \mathcal{E}_1^i(\theta^*) + \mathcal{E}_2^i(\theta^*, \theta^*) + \mathcal{E}_3^i(\theta^*, \theta^*) = 0$;

Algorithm 2: Construction of the set $\hat{\Omega}_N(p; \theta^*)$

Input: N , $\Psi(x, p; \theta^*, \mathcal{H}, n_n)$, and $\mathbb{D}_{\text{train}}$
Output: set $\hat{\Omega}_N(p; \theta^*)$
1 **Initialization:** $M_{\text{train}} = \text{length}(\mathbb{D}_{\text{train}})$ and $\hat{\Omega}_N(p; \theta^*) = \emptyset$;
2 **for** $i = 1 : M_{\text{train}}$ **do**
3 solve the learning-based CDA-RENMPC problem in (31) to
4 obtain $U^i = (u_0^i, \dots, u_{N-1}^i)$, $X^i = (x_0^i, \dots, x_N^i)$ and
5 $\hat{J}_{CDA}^*(p^i; \theta^*)$;
6 compute $\hat{L}(x_0^i, u_0^i, p^i; \theta^*)$ from (21), $\hat{d}(p^i; \theta^*)$ from (28),
7 $\hat{\alpha}(p^i; \theta^*)$ from (29), and $\hat{C}_N(p^i; \theta^*)$ from (27);
8 **if** $x_0^i \notin \hat{\Omega}_N(p; \theta^*)$ **and**
9 $\hat{J}_{CDA}^*(p^i; \theta^*) \leq \hat{L}(x_0^i, u_0^i, p^i; \theta^*) + \hat{C}_N(p^i; \theta^*)$ **then**
10 $\hat{\Omega}_N(p; \theta^*) = \hat{\Omega}_N(p; \theta^*) \cup \{x_0^i\}$;
11 **end**
12 **end**

Having identified the parameters (θ^*, θ^*) offline, we replace the intractable storage terms in (7) and (8) by the learned surrogate $\hat{\lambda}(\cdot, p; \theta^*)$, thereby obtaining the data-driven rotated costs $\hat{L}(x, u, p; \theta^*)$ in (21) and $\hat{L}_T(x, p; \theta^*)$ in (22), respectively. This yields the following learning-based reformulation of the long-horizon CDA-RENMPC problem

$$\begin{aligned}
\hat{J}_{CDA}^*(p_i; \theta^*) &= \min_{U_i, X_i} \sum_{k=0}^{N-1} \hat{L}(x_{k|t}, u_{k|t}, p_i; \theta^*) \\
&+ \hat{L}_T(x_{N|t}, p_i; \theta^*)
\end{aligned} \quad (31)$$

$$\begin{aligned}
\text{s. t. } \quad x_{0|t} &= \mathcal{M}_x p_i & k \in \mathbb{I}_0^{N-1} \\
x_{k+1|t} &= f(x_{k|t}, u_{k|t}) & k \in \mathbb{I}_0^{N-1} \\
u_{k|t} &\in \mathcal{U} & k \in \mathbb{I}_0^{N-1} \\
x_{k|t} &\in \mathcal{X} & k \in \mathbb{I}_1^N
\end{aligned}$$

The characterized domain of attraction $\hat{\Omega}_N(p; \theta^*)$ in (16) is constructed by solving problem (31) for each sample in $\mathbb{D}_{\text{train}}$ and retaining the corresponding initial states that satisfy the inclusion condition, as summarized in Algorithm 2. Consequently, $\hat{\Omega}_N(p; \theta^*)$ is a finite, sample-based representation of the domain of attraction rather than a continuous region characterized analytically.

3.4. LSTC-RENMPC

Once the cost-to-go approximation $\hat{V}(x_{1|t}, p_i; \theta^*)$ and the rotated stage cost $\hat{L}(x, u, p_i; \theta^*)$ are learned, the one-horizon RENMPC law is obtained from

$$\begin{aligned}
\hat{J}_{LSTC}^*(p_i; \theta^*, \theta^*) &= \min_{u_{0|t}, x_{0|t}, x_{1|t}} \hat{L}(x_{0|t}, u_{0|t}, p_i; \theta^*) \\
&+ x_s \hat{V}(x_{1|t}, p_i; \theta^*) \\
\text{s. t. } \quad x_{0|t} &= \mathcal{M}_x p_i \\
x_{1|t} &= f(x_{0|t}, u_{0|t}) \\
u_{0|t} &\in \mathcal{U} \\
x_{1|t} &\in \mathcal{X}.
\end{aligned} \quad (32)$$

The resulting LSTC-RENMPC controller with *Learned Storage and Terminal Cost* (LSTC) functions preserves the corresponding finite-data stability properties of the long-horizon CDA-RENMPC formulation (10), while substantially reducing the online computational burden. Algorithm 3 summarizes the corresponding online implementation procedure of the LSTC-RENMPC controller.

Algorithm 3: LSTC-RENMPC scheme

Input: $\Psi(\xi_i; \theta^*, \mathcal{H}, \mathcal{A}, \eta)$, $\Phi(p_i; \theta^*, \mathcal{H}, \mathcal{A}, \eta)$, $\epsilon > 0$
Output: u_t
1 get $p_t \in \mathcal{P}$ and obtain a pair $((p_t), u_s(p_t))$ by solving (3);
2 obtain $\hat{L}(p_t; \theta^*)$ from $\Phi(p_t; \theta^*, \mathcal{H}, \mathcal{A}, \eta)$ and evaluate
 $\hat{P}(p_t; \theta^*) = \hat{L}(p_t; \theta^*) \hat{L}^\top(p_t; \theta^*) + \epsilon I$;
3 solve problem (32) and obtain $u_{0|t}^*$;
4 apply $u_t = u_{0|t}^*$ to system (1);
5 set $t = t + 1$ and go back to 1.

4. Numerical examples

In order to demonstrate the potential of the proposed LSTC-RENMPC scheme, we consider two nonlinear CSTR benchmarks. The first example verifies that the learned storage function induces rotated stage and terminal costs that are positive definite with respect to the steady state, thereby yielding stabilizing closed-loop behavior on the characterized domain. The second example assesses closed-loop performance and confirms a Lyapunov-type decrease of the resulting optimal cost in comparison with the classical long-horizon ENMPC, while also benchmarking the online computational effort against the long-horizon CDA-RENMPC formulation.

In both demonstrative examples, nonlinear continuous-time systems were discretized using a fourth-order explicit Runge–Kutta integrator, and the corresponding nonlinear discrete-time state-space models were obtained with sampling time T_s . All ENMPC problems were formulated in CasADi [29] through its Python interface, and the corresponding automatically generated nonlinear programming (NLP) problems were solved using Ipopt [30]. The optimal steady-state pair $(x_s(p_t), u_s(p_t))$ was computed by solving the problem (3). The terminal ingredients were

Table 1
Specifications of the neural network models trained using Algorithm 1.

Case study	Network architecture		Parameters of problem (30)	
	$\Psi(\xi_i; \theta^*, \mathcal{H}, \mathcal{A}, \eta)$	$\Phi(p_i; \theta^*, \mathcal{H}, \mathcal{A}, \eta)$	Penalizing	Regularizing
CSTR-1	$\mathcal{H} = 2$	$\mathcal{H} = 2$	$\kappa_1 = 10^3$	$\epsilon = 10^{-6}$
	$\eta_h = 20, \mathcal{A}_h = \text{ReLU}, h \in \mathbb{I}_1^{\mathcal{H}}$	$\eta_h = 30, \mathcal{A}_h = \text{ReLU}, h \in \mathbb{I}_1^{\mathcal{H}}$	$\kappa_2 = 10^2$	$\gamma_1 = 10^{-4}$
	$\mathcal{A}_{\mathcal{H}+1} = \text{Linear}, \eta_{\mathcal{H}+1} = 1$	$\mathcal{A}_{\mathcal{H}+1} = \text{Linear}, \eta_{\mathcal{H}+1} = 3$	$\kappa_3 = 10^3$	$\gamma_2 = 10^{-3}$
CSTR-2	$\mathcal{H} = 4$	$\mathcal{H} = 6$	$\kappa_1 = 10^5$	$\epsilon = 10^{-6}$
	$\eta_h = 20, \mathcal{A}_h = \text{ReLU}, h \in \mathbb{I}_1^{\mathcal{H}}$	$\eta_h = 20, \mathcal{A}_h = \text{ReLU}, h \in \mathbb{I}_1^{\mathcal{H}}$	$\kappa_2 = 10^3$	$\gamma_1 = 10^{-4}$
	$\mathcal{A}_{\mathcal{H}+1} = \text{Linear}, \eta_{\mathcal{H}+1} = 1$	$\mathcal{A}_{\mathcal{H}+1} = \text{Linear}, \eta_{\mathcal{H}+1} = 10$	$\kappa_3 = 10^5$	$\gamma_2 = 10^{-2}$

Note: ReLU stands for the rectified linear unit activation function.

selected within the local terminal-design framework established in [9, Theorem 25] under the regularity and stabilizability conditions around the steady-state, and were parameterized as

$$\ell_T(x, p_i) = \frac{1}{2}(x - x_s(p_i))^T Q_T(x - x_s(p_i)) + q_T^T(x - x_s(p_i)),$$

$$u_T(x, p_i) = u_s(p_i) - K_T(x - x_s(p_i)).$$

The matrices Q_T , q_T , and K_T were obtained accordingly. To facilitate the computation of $d(p; \theta)$ in (28) and $\alpha(p; \theta)$ in (29) in the subsequent learning procedure presented in Algorithm 1, the terminal set was selected as a box contained within the corresponding admissible sublevel set. The resulting terminal ingredients, including the selected box terminal set, were numerically verified to satisfy Assumption 2 for all $p_i \in \mathcal{P}$.

To evaluate the performance of each control strategy, the controllers were simulated in N_{sim} steps from various N_{init} initial states. To compare the economic improvement of the closed loop outcomes, we consider cumulative transient profit defined as

$$\Delta J^{cl}(N_{\text{sim}}) = \sum_{i=1}^{N_{\text{sim}}} \frac{1}{N_{\text{init}}} \sum_{j=1}^{N_{\text{init}}} \left(\ell(x_s(p_i^j), u_s(p_i^j), p_i^j) - \ell(x_{0|t}, u_{0|t}^*, p_i^j) \right).$$

which quantifies how efficient each controller is in approaching the steady state.

A dataset of length $M = M_{\text{train}} + M_{\text{test}}$ was collected by performing closed-loop simulations under the classical ENMPC control law initialized from randomly generated initial states uniformly distributed within the constrained region $\mathcal{X}$. We used $M_{\text{train}} = 40000$ training samples to learn the parameters of FNN, while $M_{\text{test}} = 10000$ test samples were used to assess the performance of the learned networks $\Psi(\xi_i; \theta^*, \mathcal{H}, \eta_n)$ and $\Phi(p_i; \theta^*, \mathcal{H}, \eta_n)$ that were constructed in PyTorch [31], and trained by following the Algorithm 1 using the squared error loss function $\phi(z_1, z_2) = (z_1 - z_2)^2$ and Adam optimizer [32] with a learning rate $\alpha = 10^{-3}$ and default parameter settings.

The tuning parameters were selected empirically according to their respective roles in the learning formulation (30). In particular, an iterative *trial-and-error* procedure was used, since no dedicated theoretical tuning rule is available for the present formulation. The parameter ϵ was chosen sufficiently small to provide the required strictness and positive-definiteness margins without materially affecting the approximation objective. The regularization weights γ_1 and γ_2 were adjusted to limit excessive network parameter growth while maintaining adequate fitting accuracy. The penalty coefficients κ_1 , κ_2 , and κ_3 were tuned with regard to the relative magnitudes of the corresponding constraint-violation terms until the dissipativity, positive-invariance, and Lyapunov-decrease conditions were satisfied on both the training and test datasets, in accordance with Algorithm 1. The adopted values for both benchmark control problems are reported in Table 1.

4.1. CSTR-1: Isothermal

In order to show that the storage function $\hat{\lambda}(x, p_i; \theta^*)$ shapes positive definite functions $\hat{\mathcal{L}}(x, u, p_i; \theta^*)$ and $\hat{\mathcal{L}}_T(x, p_i; \theta^*)$ with respect to the

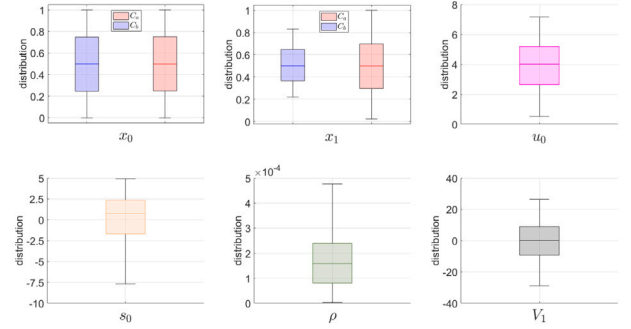


Fig. 2. Statistics of the generated data $\mathbb{D}_{\text{train}}$ summarized via boxplot.

steady state, which in turn ensures the asymptotic convergence of the trajectories resulting from the RENMPC laws in a characterized region, we consider a single first-order, irreversible chemical reaction $a \rightarrow b$ in an isothermal continuous time stirred tank reactor (CSTR) example in which the substances a and b are balanced by the following dynamic model

$$\dot{C}_a = \frac{F_r}{V_r}(C_a^f - C_a) - k_r C_a$$

$$\dot{C}_b = \frac{F_r}{V_r}(C_b^f - C_b) + k_r C_a$$

where $k_r = 0.4$ is a rate constant, C_a , C_b , $C_a^f = 1$ mol/l and $C_b^f = 0$ are the respective molar and feed concentrations of the substances a and b , F_r is a flow passing through the reactor with volume $V_r = 10$ L. The state and the manipulated input of the system are defined as $x = [C_a \ C_b]^T$ and $u = F_r$, and are limited to the box constraints $\mathcal{X} = [0, 1]^2$ and $\mathcal{U} = [0, 20]$, respectively. The nonlinear discrete-time state-space model is obtained with sampling time $T_s = 0.5$ min.

We design the classic economic control problem (2) with $N = 15$, the terminal constraint $\mathcal{X}_T(p_i) = [0.4, 0.6]^2$, and the economic stage cost

$$\ell(x, u, p_i) = ([0 \ -2]x)u + 0.5u + 0.1(u - 4)^2$$

with the associated optimal steady state pair $x_s(p_i) = [0.5, 0.5]$ and $u_s(p_i) = 4$. The statistical properties of the training data collected via closed-loop simulations of the control law under this setup are presented in Fig. 2. Leveraging this dataset, the storage function $\hat{\lambda}(x, p_i; \theta^*)$ and the cost-to-go function $\hat{\mathcal{V}}(x, p_i; \theta^*)$ were learned, where the corresponding neural network configurations and training parameters are listed in Table 1.

Fig. 3 illustrates how the learned storage function induces, through (7) and (8), positive-definite rotated stage and terminal costs with respect to the steady state. This property establishes a dissipative structure for the closed-loop dynamics, thereby ensuring stability of the resulting RENMPC scheme.

CDA-RENMPC were designed with $\hat{d}(p_i; \theta^*) = 2.47$ and $\hat{\alpha}(p_i; \theta^*) = 0.48$, resulting in $\hat{C}_{15}(p_i; \theta^*) = 35.06$. All controllers are simulated from various initial states and the corresponding state trajectories are depicted in Fig. 4, where it can be seen that the classical RENMPC and

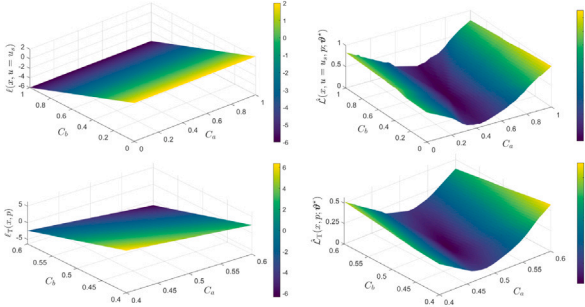


Fig. 3. Original stage and terminal costs (left) and their rotated counterparts induced by the learned storage function (right).

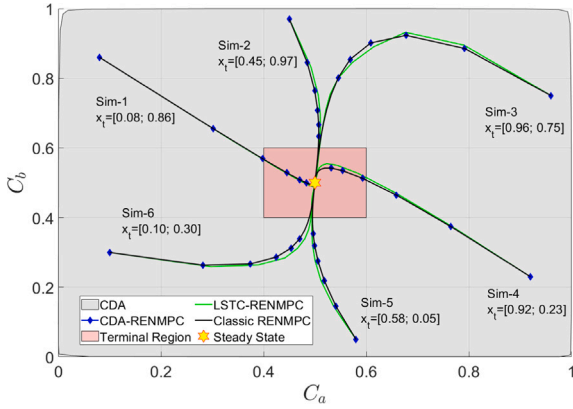


Fig. 4. State trajectories of different RENMPC schemes starting from initial state $x_{0|t}$.

CDA-RENMPC schemes yield indistinguishable trajectories, while the LSTC-RENMPC trajectories show slight discrepancies arising from the approximate data-driven cost-to-go representation $\hat{v}(x, p_t; \theta^*)$, yet they still achieve asymptotic convergence to the steady state.

4.2. CSTR-2: Two consecutive-competitive reactions

In order to assess the computational efficiency of the proposed control strategy and to show the Lyapunov property of the resulting optimal cost in comparison with the classical ENMPC scheme, we proceed by examining the dimensionless mass balancing model form of the nonlinear isothermal CSTR with two consecutive-competitive reactions [5]:

$$\begin{aligned}\dot{x}_1 &= u_1 - x_1 - \sigma_1 x_1 x_2 \\ \dot{x}_2 &= u_2 - x_2 - \sigma_1 x_1 x_2 - \sigma_2 x_2 x_3 \\ \dot{x}_3 &= -x_3 + \sigma_1 x_1 x_2 - \sigma_2 x_2 x_3 \\ \dot{x}_4 &= -x_4 + \sigma_2 x_2 x_3\end{aligned}$$

where $\sigma_1 = 1$, $\sigma_2 = 0.4$, and the system is constrained to $\mathcal{X} \times \mathcal{U} = [0, 10]^4 \times [0, 10]^2$. We consider the optimal economic control problem (2) with prediction horizon $N = 60$ and the following stage cost:

$$\ell(x, u, p_t) = -7x_3 + u_1 + u_2 + x_s \|x - x_s(p_t)\|_{Q_x}^2 + \|u - u_s(p_t)\|_{Q_u}^2,$$

where $Q_x = \text{diag}[0.1, 0.1, 0.1, 0.1]$ and $Q_u = \text{diag}[0.1, 0.1]$, and the corresponding optimal steady-state pair is $x_s(p_t) = [4.97, 1.0, 3.57, 1.44]$ and $u_s(p_t) = [10, 7.48]$. The quadratic terms are added to the economic cost to support the design and numerical construction of terminal

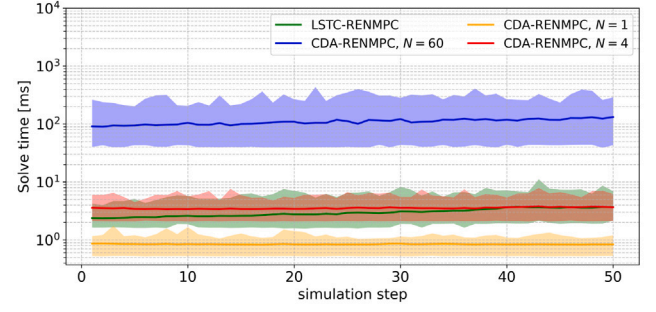


Fig. 5. Comparison of solve times for different RENMPC schemes over the simulation steps.

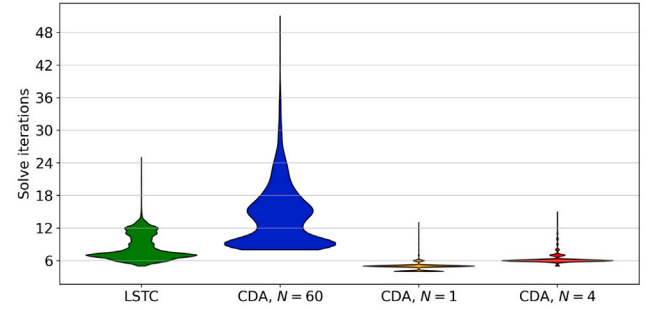


Fig. 6. Comparison of solve iterations among different RENMPC schemes.

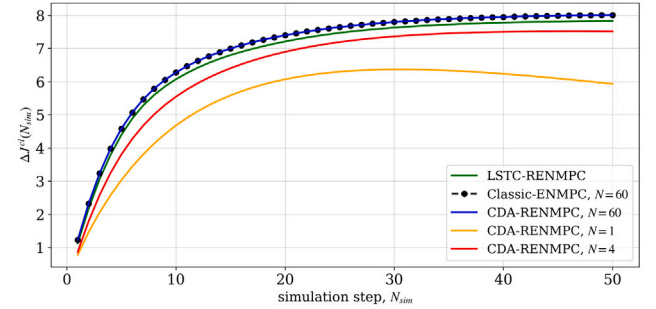


Fig. 7. Average cumulative transient profits of different ENMPC schemes.

ingredients satisfying the required assumptions. To learn the storage function $\hat{\lambda}(x, p_t; \theta^*)$ and the cost-to-go function $\hat{v}(x, p_t; \theta^*)$, we employ the networks $\Psi(\xi_t; \theta^*, \mathcal{H} = 4, n_n = 20)$ and $\Phi(p_t; \theta^*, \mathcal{H} = 6, n_n = 20)$, whose training settings are summarized in Table 1. The corresponding CDA-RENMPC scheme with $N = 60$ is constructed with $\hat{d}(p_t; \theta^*) = 22.25$ and $\hat{a}(p_t; \theta^*) = 16.84$, yielding the characterized region $\bar{\Omega}_{60}(p_t) := \{x \in \mathcal{X} : \bar{J}_{CDA}^*(p_t) \leq 1329.75\}$.

To assess the closed-loop performance and computational efficiency of the proposed scheme, all ENMPC strategies were simulated for $N_{sim} = 50$ steps from $N_{init} = 100$ initial states randomly sampled within the region $\bar{\Omega}_{60}(p_t)$.

As illustrated in Figs. 5 and 6, the proposed LSTC-RENMPC yields superior computational performance relative to the CDA-RENMPC with $N = 60$. In particular, it reduces the average computation time from 82.89 ms to 2.93 ms and achieves faster convergence, requiring only 8 iterations compared with 16 as summarized in Table 2. Furthermore, LSTC-RENMPC demonstrates improved economic performance compared to CDA-RENMPC with horizon $N = 1$ and $N = 4$ as illustrated in Fig. 7, while matching the number of decision variables and constraints

Table 2
Comparison of computational complexity among RENMPC control laws.

Controller	NLP dimension		Convergence # iterations (median)	Computational time mean [ms]
	# decision variables	# constraints		
CDA-RENMPC with $N = 60$	364	264 + 720	16	82.89
CDA-RENMPC with $N = 4$	28	20 + 48	7	3.01
CDA-RENMPC with $N = 1$	10	8 + 12	4	0.89
LSTC-RENMPC	10	8 + 12	8	2.93

Note: All computational results are obtained on a PC equipped with an Intel(R) Core(TM) i7-10700 CPU @ 2.90 GHz processor.

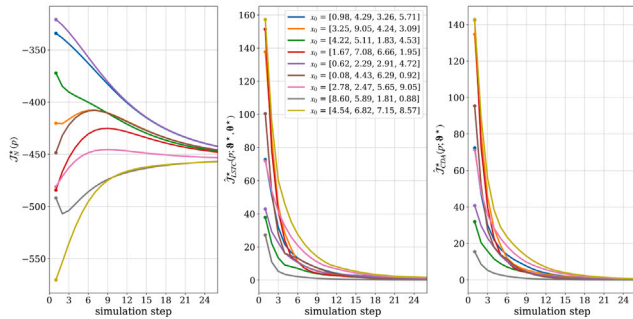


Fig. 8. Optimal cost convergence of different ENMPC schemes.

in the former case and maintaining a similar computational burden in the latter.

Beyond its computational advantages, the proposed LSTC-RENMPC generated closed-loop trajectories that remained feasible, satisfied the imposed state and input constraints, and exhibited no violations of the dissipativity, positive-invariance, or Lyapunov-decrease conditions. As illustrated in Fig. 8, the corresponding rotated optimal cost decreased monotonically, thereby confirming preservation of the Lyapunov structure of CDA-RENMPC and convergence toward the desired steady state. In contrast, the optimal cost associated with classical ENMPC attained non-positive values and exhibited fluctuations.

5. Conclusions

In this paper, we proposed a learning-based ENMPC scheme with a prediction horizon of length one by learning a storage function and a Lyapunov terminal cost approximation of the long-horizon cost-to-go. The resulting formulation preserves the stability-oriented structure of the original CDA-RENMPC problem on the sample-based characterized domain of attraction, while substantially reducing the online computational burden. Numerical case studies showed that the proposed LSTC-RENMPC reduced the average solution time from tens of milliseconds to only a few milliseconds per sampling instant, while achieving near-optimal economic performance, satisfying the imposed constraints, and exhibiting stable closed-loop behavior.

Future work will focus on establishing approximation-error bounds for the learned storage and terminal-cost functions, quantifying their impact on closed-loop performance and stability, and extending the finite-data guarantees toward continuous-state certificates.

CRediT authorship contribution statement

Shokhjakhon Abdufattokhov: Writing – original draft, Methodology, Investigation. **Mario Zanon:** Writing – review & editing, Supervision, Methodology. **Alberto Bemporad:** Writing – review & editing, Supervision, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was funded by the European Union (ERC Advanced Research Grant COMPACT, No. 101141351). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Data availability

Noexternal data used. All data is used in simulations that are reproducible by reading the paper.

References

- [1] J. Vasilj, S. Gros, D. Jakus, M. Zanon, Day-ahead scheduling and real-time economic MPC of CHP unit in microgrid with smart buildings, *IEEE Trans. Smart Grid* 10 (2019) 1992–2001.
- [2] G. Pozzato, M.A. Müller, S. Formentin, S.M. Savaresi, Economic MPC for online least costly energy management of hybrid electric vehicles, *Control Eng. Pract.* 102 (2020) 104534.
- [3] M. Soleymani, N. Bigdeli, M. Rahmani, Real-time time-varying economic nonlinear model predictive control for wind turbines, *Int. J. Electr. Power Energy Syst.* 159 (2024) 110019.
- [4] J.B. Rawlings, D.Q. Mayne, M.M. Diehl, *Model Predictive Control : Theory, Computation and Design*, Nob Hill Publishing, 2019.
- [5] J.B. Rawlings, D. Angeli, C.N. Bates, *Fundamentals of economic model predictive control*, in: 51st IEEE Conference on Decision and Control, CDC, 2012, pp. 3851–3861.
- [6] M.A. Müller, D. Angeli, F. Allgöwer, On necessity and robustness of dissipativity in economic model predictive control, *IEEE Trans. Autom. Control* 60 (2015) 1671–1676.
- [7] M. Zanon, S. Gros, A new dissipativity condition for asymptotic stability of discounted economic MPC, *Automatica* 141 (2021) 110287.
- [8] T. Faulwasser, L. Grüne, M. Müller, Economic nonlinear model predictive control: stability, optimality and performance, *Found. Trends Syst. Control.* 5 (1) (2018) 1–98.
- [9] R. Amrit, J.B. Rawlings, D. Angeli, Economic optimization using model predictive control with a terminal cost, *Annu. Rev. Control.* 35 (2011) 178–186.
- [10] M. Diehl, H. Ferreau, N. Haverbeke, Efficient numerical methods for nonlinear MPC and moving horizon estimation, *Lecture Notes in Control and Inform. Sci.* (2009) 391–417.
- [11] M. Zanon, A Gauss-Newton-like hessian approximation for economic NMPC, *IEEE Trans. Autom. Control* 66 (2020) 4206–4213.
- [12] J. Frey, K. Baumgärtner, G. Frison, M. Diehl, Multi-phase optimal control problems for efficient nonlinear model predictive control with acados, *Optim. Control. Appl. Methods* 46 (2) (2025) 827–845.
- [13] S. Pirkelmann, D. Angeli, L. Grüne, Approximate computation of storage functions for discrete-time systems using sum-of-squares techniques, *IFAC-PapersOnLine* 52 (2019) 508–513.
- [14] H. Chen, F. Allgöwer, A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability, *Automatica* 34 (10) (1998) 1205–1217.
- [15] D. Limon, T. Alamo, F. Salas, E. Camacho, On the stability of constrained MPC without terminal constraint, *IEEE Trans. Autom. Control* 51 (5) (2006) 832–836.

- [16] P. Giesl, S.F. Hafstein, Review on computational methods for Lyapunov functions, *Discrete Contin. Dyn. Syst. Ser. B* 20 (2015) 2291–2331.
- [17] D. Prokhorov, L. Feldkamp, Application of support vector machines to Lyapunov function approximation, in: *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339)*, vol. 1, 1999, pp. 383–387.
- [18] F. Berkenkamp, M. Turchetta, A.P. Schoellig, A. Krause, Safe model-based reinforcement learning with stability guarantees, in: *Neural Information Processing Systems*, 2017.
- [19] L. Grüne, Computing Lyapunov functions using deep neural networks, *J. Comput. Dyn.* 8 (2021).
- [20] N. Gaby, F. Zhang, X. Ye, Lyapunov-net: A deep neural network architecture for Lyapunov function approximation, in: *IEEE 61st Conference on Decision and Control, CDC, 2022*, pp. 2091–2096.
- [21] A.B. Kordabad, S. Gros, Q-learning of the storage function in economic nonlinear model predictive control, *Eng. Appl. Artif. Intell.* 116 (2022) 105343.
- [22] A. Romer, J. Berberich, J. Köhler, F. Allgöwer, One-shot verification of dissipativity properties from input-output data, *IEEE Control. Syst. Lett.* 3 (3) (2019) 709–714.
- [23] S. Abdufattokhov, M. Zanon, A. Bemporad, Learning convex terminal costs for complexity reduction in MPC, in: *60th IEEE Conference on Decision and Control, CDC, 2021*, pp. 2163–2168.
- [24] F. Moreno-Mora, S. Streif, Predictive control with terminal costs based on online learning using value iteration, in: *2024 European Control Conference, ECC, 2024*, pp. 1837–1842.
- [25] S. Abdufattokhov, M. Zanon, A. Bemporad, Learning Lyapunov terminal costs from data for complexity reduction in nonlinear model predictive control, *Internat. J. Robust Nonlinear Control* 34 (13) (2024) 8676–8691.
- [26] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, *Neural Netw.* 2 (1989) 359–366.
- [27] C.G. Arango, H. Asadi, L. Kooijman, C.P. Lim, Neural networks for fast optimisation in model predictive control: A review, 2023, *ArXiv abs/2309.02668*, URL: <https://api.semanticscholar.org/CorpusID:261556913>.
- [28] C.C. Aggarwal, *Neural Networks and Deep Learning: a Textbook*, Springer, Cham, Switzerland, 2018.
- [29] J. Andersson, J. Gillis, G. Horn, J. Rawlings, M. Diehl, CasADi - a software framework for nonlinear optimization and optimal control, *Math. Program. Comput.* 11 (1) (2019) 1–36.
- [30] A. Wächter, L. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Math. Program.* 106 (2006) 25–57.
- [31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, PyTorch: an imperative style, high-performance deep learning library, *Adv. Neural Inf. Process. Syst.* 32 (2019) 8024–8035.
- [32] D. Kingma, J. Ba, Adam: a method for stochastic optimization, 2014, *ArXiv abs/1412.6980*.