2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

A Novel Koopman Representation for Efficient Linear Model Predictive Control of Nonlinear Systems

Omar Sayed and Sergio Lucia

Abstract— **The Koopman operator theory is a powerful tool for the linear analysis and control of nonlinear systems that lifts the nonlinear states into a higher dimensional linear space known as the Koopman space. The linear Koopman space provides an attractive approach for designing linear control strategies for nonlinear systems. However, a significant challenge arises because the Koopman states cannot be directly related to the actual states of the system. This discrepancy can complicate the design of cost functions and constraints for a model predictive controller. In this work, we introduce a novel Koopman representation combined with a training scheme that resolves this issue by defining auxiliary states that are injective and monotonic to the original states. We evaluate the effectiveness of the proposed scheme through numerical experiments.**

I. Introduction

In recent years, Model Predictive Controllers (MPC) have gained popularity due to their versatility. This is primarily attributed to their ability in handling cost functions and constraints which are tailored for various applications [1], [2]. Moreover, MPC excels at managing a wide spectrum of control challenges, including nonlinear systems and effectively dealing with uncertainties [3], [4].

However, an MPC scheme relies on a prediction model to simulate future states and then implicitly determine the optimal control law. This can be challenging when dealing with nonlinear models due to computational complexity and tractability issues [5], necessitating the use of simplifications such as linearization [6].

A promising idea to reduce the computational complexity associated with nonlinear models is the use of Koopman operator theory [7], [8]. Koopman theory states that by a nonlinear transformation of states to an infinite dimension, it is possible to linearly represent the dynamics of a nonlinear system in what is known as the Koopman embedding space. For autonomous systems, researchers have worked to make this theory practical by seeking finite approximations of the Koopman space for autonomous nonlinear systems [9], [10]. Extensions have also been explored for systems with exogenous inputs [11], [12], [13].

However, finding the appropriate transformation function has its own challenges especially when knowledge of the underlying process is limited [14]. This has led research towards the use of deep neural networks (DNNs) to approximate lifting functions for autonomous systems [15], [16], [14], and for systems with exogenous inputs [17], [18], [19].

*This work was not supported by any organization

Omar Sayed and Sergio Lucia are with the Chair of Process Automation Systems, TU Dortmund University, Dortmund, Germany {omar.sayed, sergio.lucia}@tu-dortmund.de

An issue that arises when attempting to use the lifted system as a prediction model with MPC is the loss of interpretability between the Koopman states and the original state space, since the Koopman states have no physical meaning. This challenge becomes apparent when designing state constraints and cost functions for MPC, as discussed in [19] and [20].

Several approaches have been proposed to address these challenges. One approach is to use a linear inverse transformation to establish a relationship between the Koopman space and the original state space, as in [17] and [21], thus enabling linear MPC. Another approach is to directly include nonlinear states in the Koopman space, which is a special case of the former, as in [6], [20] and [22]. However, these methods may result in reduced prediction accuracy as they rely on the assumption of a linear relationship between the two spaces.

The main contribution of this work is as follows: First, we propose a novel modelling and control scheme to impose constraints and cost function design directly in the Koopman space by introducing auxiliary states. These auxiliary states are injective and monotonic with respect to the original states, which is achieved through a specific network structure and training scheme outlined in this work. Secondly, the proposed scheme is evaluated against a baseline architecture and controller in a numerical example, testing its performance with different cost functions and state constraints.

This paper is structured as follows: Section II, gives a brief introduction to Koopman theory and its extension to systems with inputs. Section III explores the Koopman operators in the context of DNNs and presents our proposed network structure and training scheme. In section IV, we illustrate the design of MPC using linear Koopman models, considering both the baseline and the proposed approach. In section V, we evaluate and compare the performance of our proposed algorithms with the baseline methods. Finally, we conclude our work in section VI.

II. Background

A. Koopman theory for autonomous systems

Consider the following discrete dynamical system:

$$x_{k+1} = F(x_k), \tag{1}$$

where $x_k \in \mathbb{R}^{n_x}$ represents the state vector of the system at time step k, with a dimension n_x. The function $F : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ describes the nonlinear dynamics and simulates the states of the system by a time step t_s.

The Koopman operator acts on a lifting function $g : \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$, which nonlinearly transforms the original states of dimension n_x into the Koopman space of dimension n_z, which is often considered to be infinite-dimensional. This transformation allows the system's dynamics to be represented linearly by the Koopman operator $\mathcal{K}$, so that the following relation holds:

$$\mathcal{K}g(x_k) = g(x_{k+1}) = g \circ F(x_k). \tag{2}$$

To approximate $\mathcal{K}$ and g in practice, finite dimensional approximations are derived using time series data sampled from the original nonlinear system. The goal is to find a finite embedding space that is typically larger than the true state space of the original system. For a given dataset $D = [x_0, x_1, \ldots, x_N]$ and a chosen dimension of the Koopman space n_z this approximation can be obtained by an optimization problem given by

$$\underset{g,\mathcal{K}}{\operatorname{argmin}} \sum_{k=0}^{N-1} \|g(x_{k+1}) - \mathcal{K}g(x_k)\|. \tag{3}$$

B. Koopman operator with controls

In this work, we consider a controlled system represented as $x_{k+1} = F(x_k, u_k)$. To adapt the Koopman theory to such systems, following the approach of previous studies such as [17], [11], and [12], we redefine the equation (2) as:

$$\mathcal{K}(g(x_k), u_k) = g(F(x_k, u_k)) = g(x_{k+1}), \tag{4}$$

where $F : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ represents the dynamic equation, considering both the state x_k and the control input $u_k \in \mathbb{R}^{n_u}$ with dimension n_u. The Koopman operator, is decomposed into two components: $\mathcal{K}_x \in \mathbb{R}^{n_z \times n_z}$ and $\mathcal{K}_u \in \mathbb{R}^{n_z \times n_u}$, both of which are linear functions learned for the state and control input.

The lifting function and the Koopman operator for a given dimension of the Koopman space n_z, can be obtained by solving the following optimization problem:

$$\underset{g,\mathcal{K}_x,\mathcal{K}_u}{\operatorname{argmin}} \sum_{k=0}^{N-1} \|g(x_{k+1}) - (\mathcal{K}_x g(x_k) + \mathcal{K}_u u_k)\|. \tag{5}$$

Following from (5) a nonlinear system can be represented in the Koopman space as:

$$g(x_{k+1}) = \mathcal{K}_x g(x_k) + \mathcal{K}_u u_k, \tag{6}$$

$$z_{k+1} = \mathcal{K}_x z_k + \mathcal{K}_u u_k. \tag{7}$$

In this context, $\mathcal{K}_x$ and $\mathcal{K}_u$ can be associated with the state matrix A and the input matrix B resembling their roles in linear control theory. For more indepth information on the Koopman operator, readers can refer to [23].

III. Deep Koopman

In this section, we review Koopman operators in the context of deep neural networks. We explore methods involving nonlinear transformations to and from the Koopman space using autoencoders [24]. We present both the standard baseline method and our proposed methods for comparison.

A. Standard Data-based Koopman Method as Baseline

Learning the Koopman operator usually consists of two steps. The first is to lift the states. The second is to learn the linear dynamics for the lifted states. This can be achieved simultaneously by the following single-step loss:

$$\mathcal{L}_{\text{ss}} = \sum_{k=0}^{N-1} \|x_{k+1} - \psi^{-1}(\mathcal{K}_x \psi(x_k) + \mathcal{K}_u u_k))\|, \tag{8}$$

where $\psi : x \in \mathbb{R}^{n_x} \to z \in \mathbb{R}^{n_z}$ and $\psi^{-1} : z \to x$ denote the encoder and decoder. Typically, they are chosen as neural networks $\mathcal{N}(.;\theta)$ with trainable parameters θ. $\mathcal{K}_x$ and $\mathcal{K}_u$, are parameterized linear matrices, corresponding to the system and input matrices. Since our goal is to use these models as a prediction model in MPC, it is important to train them also on a multi-step loss to reduce accumulation of errors when performing multi-step predictions [25], [26]. For this we define the following k-step loss:

$$\mathcal{L}_{\text{ms}} = \sum_{p=0}^{N-k} \sum_{m=p+1}^{p+k} \|\mathbf{X}_{p,m} - \psi^{-1}(\hat{\mathbf{X}}_{p,m})\|, \tag{9}$$

$$\text{where,} \quad \mathbf{X}_{p,m} := [x_p, x_{p+1}, \ldots, x_m], \tag{10}$$

$$\hat{\mathbf{X}}_{p,m} := [\hat{x}_{[p-1,p]}, \hat{x}_{[p-1,p+1]}, \ldots, \hat{x}_{[p-1,m]}], \tag{11}$$

$$\hat{x}_{[p,m]} = \mathcal{K}_x^{m-p} \psi(x_p) + \sum_{i=p}^{m-1} \mathcal{K}_x^{m-i-1} \mathcal{K}_u u_i, \tag{12}$$

where $\hat{x}_{[p,m]}$ denotes the state at time step m using an initial state at time p and simulating the dynamics using the input trajectory from time step p up to m. In this case, the number of k-steps prediction for $\hat{x}_{[p,m]}$ is $(k = m - p)$.

B. Proposed Architecture

Our proposed scheme involves assigning specific auxiliary states in the Koopman space and imposing constraints on these states during training to ensure their injectivity and monotonicity with respect to the original states. The importance of these constraints will become apparent when we formulate the MPC problem in section IV. A nonlinear transformation, denoted as $\psi : X \in \mathbb{R}^{n_x} \to [Z \in \mathbb{R}^{n_z}, S \in \mathbb{R}^{n_x}]$, maps to two distinct domains. Here, Z denotes the space of Koopman states, which carries the linear dynamics of the nonlinear system, while S denotes the space of auxiliary states, with a dimensionality identical to that of the original state space. We consider the following assumption:

Assumption 1: Let $f(\cdot;\theta) : x \in X \to s \in S$ and its inverse $f^{-1}(\cdot;\theta) : s \to x$ represent one-dimensional mapping functions between two domains. For a trajectory $T = [x_0, x_1, ..., x_n]$ spanning f over $[a, b]$ and $f(T)$ spanning f^{-1} over $[c, d]$, there exists a set of parameters θ such that $\min_\theta \|T - f(f(T;\theta);\theta)^{-1}\| = 0$.

The assumption is that an encoding and decoding function can be learned for a one-dimensional data set, with a latent space equal to the original space and a domain of validity defined by the data set. This can be achieved if the data is rich, covering the full range of validity, and the structure of the function is of appropriate complexity. $f(.)$ can be a

trivial function, but we want to impose constraints on the transformation, as we will see later.

Lemma 1: Let Assumption 1 hold, then functions f and f^{-1} are bijective between the domains of $X \in [a, b]$ and $S \in [c, d]$. Furthermore, since both f and f^{-1} are one-dimensional and bijective, it follows that they are strictly monotonic.

Proof: See [27]. ∎

In Theorem 1 we look to expand this Lemma to a multi-dimensional neural network. We make use of the Hadamard product $\odot$ that denotes the elementwise multiplication.

Theorem 1: Consider a masking vector $M \in \mathbb{R}^n$ defined as $M_i := [0, \cdots, 1, \cdots, 0]$, where it contains a single unitary element only in the i-th position. Then for a feedforward neural network $\mathcal{N}(\cdot;\theta) : \mathbb{R}^n \to \mathbb{R}^n$, applying the filter during training and testing to the input and the output $M_i \cdot \mathcal{N}(M_i \odot x)$ will result in a one-to-one mapping between the i-th input and the i-th output, so Lemma 1 will hold for the i-th dimension.

Proof: Consider an input vector $[x_1, \cdots, x_n]$, and a neural network with a single hidden neuron activated by ReLU function. The output of the hidden neuron when applying the mask M_i is as follows:

$$h_1 = \max(0, [w_{11} \cdots, w_{1i}, \cdots, w_{1n}] \begin{bmatrix} x_1 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix} \odot \begin{bmatrix} 0 \\ \vdots \\ 1_i \\ \vdots \\ 0 \end{bmatrix} + b_1),$$

$$= \max(0, w_{1i} x_i + b_1)$$

Building on this, the output layer can be expressed as $\mathbf{O} = [w_{21} h_1, \cdots, w_{2n} h_1]$, where $[w_{21}, \cdots, w_{2n}]$ represent the weights at the output layer. Applying M_i to the output layer (i.e. $M_i \cdot \mathbf{O}$) results in $\mathbf{O}_i = f(x_i)$, where the function f includes the weights multiplication, bias addition, and ReLU activation applied to the input x_i. Thus using M_i during training and testing is equivalent to having a neural network with an input-output dimension of one $\tilde{\mathcal{N}} : \mathbb{R}^1 \to \mathbb{R}^1$ mapping x_i to $\mathbf{O}_i$. ∎

The validity of Theorem 1 is independent of the depth or width of the neural network. Furthermore, the mask M_i can be used for different values of i in multiple feed-forward instances for the same neural network giving a monotonic and injective relation between multiple inputs and outputs. Theorem 1 could be seen as equivalent to learning n separate networks, one for each input. However, in larger networks, a significant portion of these weights is shared, especially in the hidden layers. This sharing of weights helps in learning a stable and more general transformation. Moving on to applying Theorem 1 on the auxiliary states $s_i \in S$ with $M_i \in \mathbb{R}^{n_z + n_x}$ and $\tilde{M}_i \in \mathbb{R}^{n_x}$, we define the following reconstruction loss.

$$L_{\text{aux}_1} = \sum_{k=0}^{N-1} \sum_{i=0}^{n_x} \| x_{i,k} - \psi^{-1}(s_i)_i \|, \tag{13}$$
$$s_i := \psi(x_k) \odot M_i, \qquad \psi^{-1}(.)_i := \tilde{M}_i \cdot \psi^{-1}(\cdot),$$

where $x_{i,k}$ is a scalar representing the i-th state at time step k. A general remark is that by applying the mask constraint to the decoder, we implicitly apply it to the encoder, since both functions are related in terms of performance and backpropagation [28]. Following this, we can reformulate the single-step and multi-step losses as

$$L_{\text{aux}_2} = \sum_{k=0}^{N-1} \sum_{i=1}^{n_x} \| x_{i,k+1} - \psi^{-1}(\tilde{s}_i)_i \|, \tag{14}$$
$$\tilde{s_i} := M_i \odot (C\mathcal{K}_x \psi(x_{i,k}) + C\mathcal{K}_u u_i),$$
$$L_{\text{aux}_3} = \sum_{p=0}^{N-1} \sum_{m=p+1}^{p+k} \sum_{i=1}^{n_x} \| (\tilde{M}_i^T \cdot X_{p,m} - \psi^{-1}(\hat{s}_i)_i) \|, \tag{15}$$
$$\hat{s}_i := M_i \odot C\hat{X}_{p,m},$$

where, $\hat{X}$ and X are computed using (10) and (11) respectively. Since the Koopman states are linear, we can relate them to the auxiliary states linearly as $s_k = Cz_k$ without loss of generalizability. This is due to the single-step and multi-step loss constraints, which already force the auxiliary states to be observable to the latent states.

Furthermore, it can be argued that L_{aux_2} is already a part of L_{aux_3}, but in practice it shows better training results when both losses are used. The total loss function is the weighted sum of the multi-step loss L_{ms}, for the Koopman states (Z) and the three auxiliary states (S) losses L_{aux_1}, L_{aux_2} and L_{aux_3}. A schematic of the architecture is shown in Fig. 1.

IV. MPC in the Koopman Space

MPC relies on a prediction model and a specific cost function to compute the input trajectory that optimizes a customized cost function over a given horizon H. For a nonlinear system of the form $x_{k+1} = F(x_k, u_k)$, a nonlinear MPC can be formulated as follows:

$$\min_{\tilde{u}} \quad \sum_{k=0}^{H-1} x_k^T Q x_k + u_k^T R u_k, \tag{16}$$
$$\text{s.t.} \qquad x_{k+1} = F(x_k, u_k), \;\; x_0 = x_{\text{init}},$$
$$x_{\text{lb}} \le x_k \le x_{\text{ub}}, \qquad u_{\text{lb}} \le u_k \le u_{\text{ub}},$$

where, for simplicity we choose a quadratic cost function defined by the weighting matrices $Q \in \mathbf{R}^{n_x \times n_x}$, $R \in \mathbf{R}^{n_u \times n_u}$. The lower and upper bounds of the states are denoted by x_{lb}, $x_{\text{ub}} \in \mathbf{R}^{n_x}$ and u_{lb}, $u_{\text{ub}} \in \mathbf{R}^{n_u}$ are the input constraints. The solution to this optimization problem is the optimal input sequence, $\tilde{u}$, that minimizes the cost function while satisfying the constraints.

The optimization problem (16) is non-convex and computationally complex. To address this issue, an alternative approach is to leverage the Koopman states to reformulate the nonlinear optimization into a convex form. If the cost

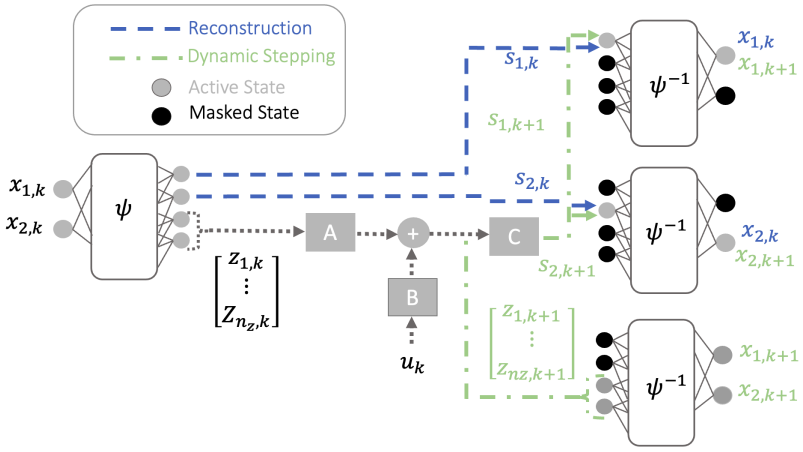

Fig. 1. Proposed schematic: In here, ψ^{-1} corresponds to the same decoder used across multiple feed-forward instances. The blue lines denote the reconstruction of the original states from the axillary states. During this step, the original state is reconstructed directly from the auxiliary state for the same time step k, without incorporating dynamic propagation. The green lines denotes the states propagated by one time step. An essential feature of this architecture is the masking of the decoder input to establish a one-to-one relationship between the auxiliary states and the original states. For instance, the top right decoder is compelled to map the first auxiliary state s_1 to x_1.

function is quadratic and the constraints are linear (16) can be reduced into a quadratic convex problem as in [17], [21] to be:

$$\begin{aligned} \min_{\tilde{u}} \sum_{k=0}^{H-1} z_k^T \hat{Q} z_k + u_k^T R u_k, \qquad (17) \\ \text{s.t.} \quad z_{k+1} = \mathcal{K}_x z_k + \mathcal{K}_u u_k, \\ z_0 = \psi(x_{\text{init}}), \;\; x_{\text{lb}} \leq \hat{C} z_k \leq x_{\text{ub}}, \;\; u_{\text{lb}} \leq u_k \leq u_{\text{ub}}, \end{aligned}$$

where $\hat{C}$ is a linear matrix defined as $\hat{C} := X\psi(X)^\dagger$ and $\hat{x}_k = \hat{C} z_k$. Here, $\dagger$ donates the Moore-Penrose pseudoinverse and X are state trajectories from the training dataset. The state cost matrix Q is transformed for the Koopman states as $\hat{Q} = \hat{C}^T Q \hat{C}$. A clear disadvantage of this formulation is that it is assumed that the nonlinear decoder function ψ^{-1} can be replaced by a linear relation $\hat{C}$. This can hinder the performance of the MPC controller as it will be shown in Section V.

We propose to directly formulate the MPC optimization (16) in the Koopman space without relying on inverse transformations. This can be facilitated by using the injective and monotonic auxiliary states defined in section III-B. We can reformulate (16) as:

$$\begin{aligned} \min_{\tilde{u}} \quad \sum_{k=0}^{H-1} s_k^T \tilde{Q} s_k + u_k^T R u_k, \qquad (18) \\ s.t. \;\; z_{k+1} = \mathcal{K}_x z_k + \mathcal{K}_u u_k, \;\; s_{k+1} = [C\mathcal{K}_x \;\; C\mathcal{K}_u] \begin{bmatrix} z_k \\ u_k \end{bmatrix}, \\ z_0 = \psi(x_{\text{init}}), \;\; s_{\text{lb}} \leq s_k \leq s_{\text{ub}}, \;\; u_{\text{lb}} \leq u_k \leq u_{\text{ub}}. \end{aligned}$$

In this formulation C, is different from $\hat{C}$ as it relates the Koopman states to the auxiliary states and operates within a linear Koopman space. s_{lb} and s_{ub} are the auxiliary state constraints which can be related to the true states of the system as

$$s_{\text{lb}}, s_{\text{ub}} = \begin{cases} \psi(x_{\text{lb}}), \psi(x_{\text{ub}}) & \text{if } \psi(x_{\text{lb}}) < \psi(x_{\text{ub}}) \\ \psi(x_{\text{ub}}), \psi(x_{\text{lb}}) & \text{otherwise,} \end{cases}, \qquad (19)$$

where this condition stems from the fact that the relationship between the auxiliary states and the original states can be either monotonically decreasing or increasing. In a similar fashion, we can relate $\tilde{Q}$ to Q by the element-wise operation as

$$\tilde{Q}_{i,j} = \frac{(Q_{ij} - x_{lb,j})(s_{ub,j} - s_{lb,j})}{x_{ub,j} - x_{lb,j}} + s_{lb,j}. \qquad (20)$$

It is important to emphasize that the relation between $\tilde{Q}$ and Q is not an equivalent transformation, but maintains an optimization direction. The exploration of alternative representations of $\tilde{Q}$ will be the subject of future work.

V. Results

We evaluate the performance of both the proposed and baseline methods in the context of MPC control under two different scenarios: setpoint tracking and economic cost. All of our experiments consider the open-loop application of a linear MPC in order to better analyze the prediction quality of the underlying models. Consequently, optimal control trajectories are computed using the formulations in (17) and (18) by solving the optimization problem once and then applying the control trajectory directly to the nonlinear system. We evaluate the schemes in terms of constraint satisfaction and cost function performance.

A. Continuously Stirred Tank Reactor

Consider the following two-state nonlinear CSTR, adapted from [29]:

$$\begin{aligned} \dot{C}_A &= \frac{F}{V_r}(C_{A_0} - C_A) - k_0 e^{\frac{-E}{RT_r}} C_A^2, \\ \dot{T}_r &= \frac{F}{V_r}(T_0 - T_r) - \frac{\Delta H}{\rho C_p} k_0 e^{\frac{-E}{RT_r}} C_A^2 + \frac{\dot{Q}}{\rho C_P V_r}, \qquad (21) \end{aligned}$$

where the states of the system are C_A the concentration of the reactant A, and T_r the temperature of the reactor. These equations model an irreversible exothermic reaction $A \rightarrow B$, in which the reactant A is converted to the reactant B, releasing heat in the process. The temperature of the reactor is controlled by the manipulated variable $\dot{Q}$, the rate of heat input to the system. C_{A_0} and T_0 are constants and denote the initial concentration of reactant A and the initial temperature of the reactor. F is the flow rate of reactant A into the reactor and is fixed at 2 m^3/hr. The rest of the parameters are constants and are used as in [29].

The goal is to learn a linear representation of (21) under Koopman using the proposed and the baseline schemes. A training data set consisting of 500 trajectories was collected by simulating (21) with uniform random step inputs at different time steps in the range $-2000 \leq Q \leq 8000$ following a similar setup as in [30].

The proposed network architecture consists of an encoder and a decoder with two hidden layers of 64 and 32 neurons, respectively, and ReLU activation functions. To balance between the latent space size and prediction accuracy, a latent space size of $n_z = 7$ was chosen. These states dynamically simulate the model for 100 steps of composed linear predictions. In addition, two states are designated as auxiliary states. The baseline network architecture is similar to the proposed architecture without the auxiliary states.

To assess the quality of the learned models the proposed and the baseline models were tested against a control input consisting of five steps within the range of the training input. The parameters of the $\hat{C}$ matrix, for the baseline model, were computed by least squares such that $\hat{C} = X\psi(X)^{\dagger}$ using 200 trajectories from the training dataset. It was shown that adding an additional loss $\min\|X - \hat{C}\psi(X)\|$ during training resulted in better performance for the baseline. For the proposed scheme, we propagate the dynamics using the Koopman states and then reconstruct the original states using only the auxiliary states. The results are shown in Fig. 2.

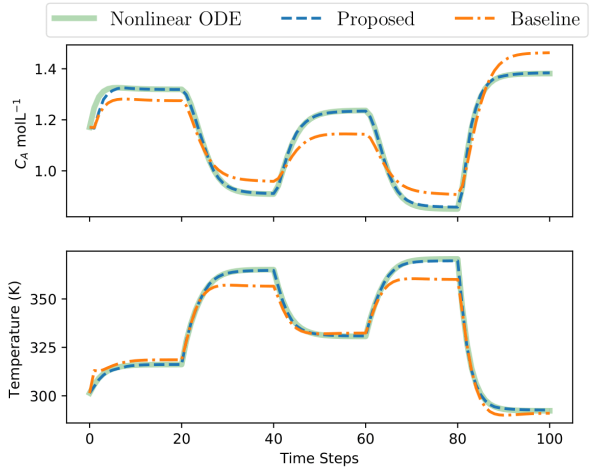

Fig. 2. CSTR modelling: Nonlinear ODE here resembles the true system response to the varying step input.

The proposed method leads to better performance due to the architecture that does not assume a linear decoder, but directly relates the states in Koopman space to the real states through the auxiliary states using the nonlinear decoder. We proceed to design an MPC controller on both models. In the first scenario, we evaluate the performance of the trained models for setpoint tracking. Specifically, we employ the MPC to control the concentration of reactant A in the reactor. This is achieved by manipulating the temperature of the reactor through the rate of heat input $\dot{Q}$. The results are visualized in Fig. 3.

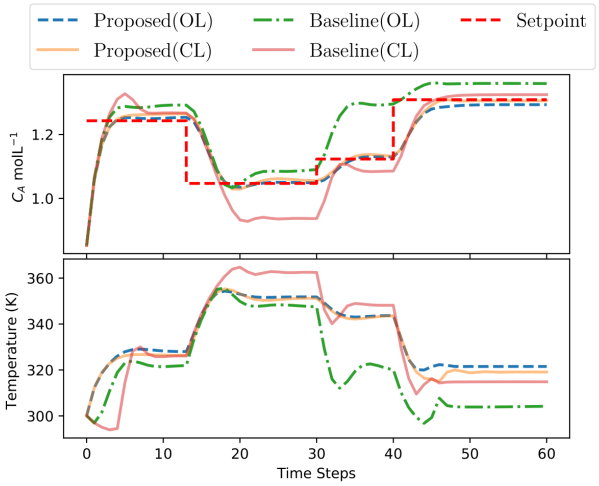

Fig. 3. CSTR setpoint tracking: Baseline and proposed are the nonlinear system simulated with control trajectories from both the proposed and baseline linear MPC, in both the open-loop (OL) and closed-loop (CL) settings.

The results show that the proposed method successfully tracks the setpoints, with only a small steady-state error. On the other hand, it can be observed that the strong assumption of a linear relationship between the Koopman states and the auxiliary states does not hold. This is evident from the steady state error of the baseline controller. Operating the baseline controller in closed-loop reduces the steady state error between $0 \leq t \leq 13$ and $30 \leq 60$, but the proposed method shows better results in both open and closed-loop settings.

In the second scenario we test our proposed method and baseline using an economic cost to reduce the amount of C_A in the reactor. This results in a higher product of reactant C_B. We further investigate the behaviour of the MPC optimization on handling constraints for T_r on two levels. The results for the open-loop simulation only are shown in Fig. 4 for feasibility reasons, as we are operating close to the constraints.

The results show that the baseline model achieves a lower concentration for C_A, indicating a better performance compared to the proposed scheme. However, when consider constraint violation on T_r, we can see that the baseline model violates the constraints in both settings. This observation highlights the ability of the proposed scheme to satisfy the state constraints.

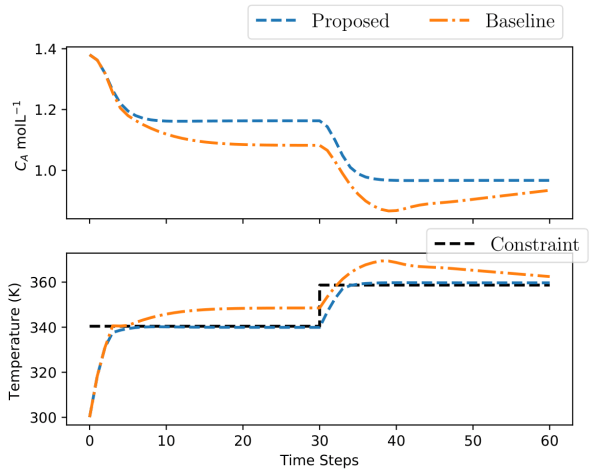

Fig. 4. CSTR using economic cost: Baseline and proposed are the nonlinear system simulated with control trajectories from both the proposed and baseline linear MPC.

VI. Conclusion

This work presents a network structure and a training scheme specifically tailored to overcome the inherent difficulties in formulating constraints and designing cost functions for the linear Koopman states when employed for model predictive control. The key innovation is to eliminate the reliance on linear approximations, which is achieved by using auxiliary states within the Koopman space. These auxiliary states have the desirable properties of being injective and monotonic with respect to the original states, thus allowing direct control design within the Koopman space. The results show that the proposed scheme outperforms standard methods in terms of both control performance and constraint satisfaction. Our future work will extend the proposed approach to larger case studies and to consider systems with unmeasured states.

References

[1] James Blake Rawlings, David Q Mayne, and Moritz Diehl. *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI, 2017.

[2] Max Schwenzer, Muzaffer Ay, Thomas Bergs, and Dirk Abel. Review on model predictive control: An engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5-6):1327–1349, 2021.

[3] Sergio Lucia, Tiago Finkler, and Sebastian Engell. Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. *Journal of process control*, 23(9):1306–1319, 2013.

[4] Sergio Lucia and Benjamin Karg. A deep learning-based approach to robust nonlinear model predictive control. *IFAC-PapersOnLine*, 51(20):511–516, 2018.

[5] S Joe Qin and Thomas A Badgwell. An overview of nonlinear model predictive control applications. *Nonlinear model predictive control*, pages 369–392, 2000.

[6] Yusuke Igarashi, Masaki Yamakita, Jerry Ng, and H Harry Asada. Mpc performances for nonlinear systems using several linearization models. In *2020 American Control Conference (ACC)*, pages 2426–2431. IEEE, 2020.

[7] Bernard O Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

[8] Bernard O Koopman and J v Neumann. Dynamical systems of continuous spectra. *Proceedings of the National Academy of Sciences*, 18(3):255–263, 1932.

[9] Clarence W Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, and Dan S Henningson. Spectral analysis of nonlinear flows. *Journal of fluid mechanics*, 641:115–127, 2009.

[10] Matthew O Williams, Maziar S Hemati, Scott TM Dawson, Ioannis G Kevrekidis, and Clarence W Rowley. Extending data-driven koopman analysis to actuated systems. *IFAC-PapersOnLine*, 49(18):704–709, 2016.

[11] Xu Ma, Bowen Huang, and Umesh Vaidya. Optimal quadratic regulation of nonlinear system using koopman operator. In *2019 American Control Conference (ACC)*, pages 4911–4916. IEEE, 2019.

[12] Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, 2018.

[13] Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Generalizing koopman theory to allow for inputs and control. *SIAM Journal on Applied Dynamical Systems*, 17(1):909–930, 2018.

[14] Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10), 2017.

[15] Enoch Yeung, Soumya Kundu, and Nathan Hodas. Learning deep neural network representations for koopman operators of nonlinear dynamical systems. In *2019 American Control Conference (ACC)*, pages 4832–4839. IEEE, 2019.

[16] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

[17] Yiqiang Han, Wenjian Hao, and Umesh Vaidya. Deep learning of koopman representation for control. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1890–1895. IEEE, 2020.

[18] Jan C Schulze and Alexander Mitsos. Data-driven nonlinear model reduction using koopman theory: Integrated control form and nmpc case study. *IEEE Control Systems Letters*, 6:2978–2983, 2022.

[19] Akhil Ahmed, Ehecatl Antonio del Rio-Chanona, and Mehmet Mercangöz. Linearizing nonlinear dynamics using deep learning. *Computers & Chemical Engineering*, 170:108104, 2023.

[20] Haojie Shi and Max Q-H Meng. Deep koopman operator with control for nonlinear systems. *IEEE Robotics and Automation Letters*, 7(3):7700–7707, 2022.

[21] Zuowei Ping, Zhun Yin, Xiuting Li, Yefeng Liu, and Tao Yang. Deep koopman model predictive control for enhancing transient stability in power grids. *International Journal of Robust and Nonlinear Control*, 31(6):1964–1978, 2021.

[22] Aqib Hasnain, Nibodh Boddupalli, Shara Balakrishnan, and Enoch Yeung. Steady state programming of controlled nonlinear systems via deep dynamic mode decomposition. In *2020 American Control Conference (ACC)*, pages 4245–4251. IEEE, 2020.

[23] Steven L Brunton, Marko Budišić, Eurika Kaiser, and J Nathan Kutz. Modern koopman theory for dynamical systems. *arXiv preprint arXiv:2102.12086*, 2021.

[24] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49. JMLR Workshop and Conference Proceedings, 2012.

[25] H-T Su and TJ McAvoy. Neural model predictive control of nonlinear chemical processes. In *Proceedings of 8th IEEE International Symposium on Intelligent Control*, pages 358–363. IEEE, 1993.

[26] Omar Sayed and Sergio Lucia. Recursive least squares-based identification for multi-step koopman operators. *(in press) 2024 European Control Conference (ECC)*, 2024.

[27] Kenneth George Binmore. *Mathematical Analysis: a straightforward approach*. Cambridge University Press, 1982.

[28] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[29] Anas Alanqar, Matthew Ellis, and Panagiotis D Christofides. Economic model predictive control of nonlinear process systems using empirical models. *AIChE Journal*, 61(3):816–830, 2015.

[30] Jan C Schulze, Danimir T Doncevic, and Alexander Mitsos. Identification of mimo wiener-type koopman models for data-driven model reduction using deep learning. *Computers & Chemical Engineering*, 161:107781, 2022.