2022 American Control Conference (ACC)
Atlanta, USA, June 8-10, 2022

Data-Driven Control: Overview and Perspectives*

Wentao Tang[1] and Prodromos Daoutidis[2]

Abstract— **Process systems are characterized by nonlinearity, uncertainty, large scales, and also the need of pursuing both safety and economic optimality in operations. As a result they are difficult to control effectively. Data-driven techniques such as machine learning algorithms can provide complementary tools and insights to classical model-based control by enhancing the capability of modeling the dynamics of complex systems and the maintenance of control performance. Moreover, by learning the behavior of plants and controllers as black boxes, data-driven techniques can enable a completely model-free control paradigm. Hence, data-driven process control has the potential to mitigate the challenges of state-of-the-art control technology and yield generic, adaptive, and scalable strategies. This paper aims at providing an overview and conceptual classification of the main approaches in this emerging and promising field, and identifying current limitations and future directions.**

I. Introduction

The acquirement, processing, and analysis of "big data" has become a prevalent topic in science and engineering, also reviving the interest of using data-driven techniques for automatic control [1], [2]. In this paper, we are interested in the control of *process systems*, namely process control. Examples of process systems include oil refineries, chemical plants, pharmaceutical and fermentation processes, and semiconductor manufacturing systems. The macroscopic and phenomenological dynamics of process systems are determined by the transport of mass, energy, and momentum subject to restrictions imposed by thermodynamic laws [3]. Motivated by the growing capability of collecting datasets of high volume and variety from the operations of process systems, machine learning techniques have been extensively adopted in the context of process data analytics [4], [5].

A. Model-based process control and its limitations

The strategy of controlling process systems based on their dynamic models, i.e., *model-based control*, was largely motivated by the growing availability of first-principles models and the emergence of state-space control theory. The exploration of applying optimal control to chemical processes started in the 1960s [6]–[8] and culminated in the extensive implementation of model predictive control (MPC) technology since the 1980s [9], [10]. The corresponding optimal control problems are formulated by taking some economic costs and/or deviations of the predicted trajectories from reference ones as the objective function, and incorporating the dynamic model and operating limits as constraints. These optimization problems are formulated and solved in a repetitive manner based on real-time measurements and estimations (typically using Kalman filters) of the plant states [11].

The theoretical stability properties of MPC in different formulations was established through extensive studies in the 1990s [12]. Since then, many variants of MPC, e.g., nonlinear MPC [13], economic MPC [14], distributed MPC [15], and fast MPC [16] have been developed based on the underlying optimization theory. In parallel to MPC, several other model-based strategies have been studied in the process control community, especially robust control for linear systems with uncertainty [17] and input-output linearizing control for nonlinear dynamics [18], and the conceptual relations between these strategies have been elucidated in a number of studies [19], [20].

Model-based control strategies, however, may fall short due to the complexity of process systems. Importantly, the establishment of models that are both accurate and easy to use for model-based control can be challenging. Since first-principles models are often complex and thus difficult to derive and reduce, practically, dynamic models can be obtained through system identification based on intentional perturbations of the process [21], which is often time-consuming and requires extensive human interventions to guarantee the model quality. The current system identification procedures are often restricted to model structures that are mostly linear and deterministic. While linear models are often claimed to be satisfactory for processes operated in small regions around steady states, their resulting control performance can deteriorate when the operation modes vary significantly, in which case nonlinear controllers must be deployed. Many process systems related to pharmaceuticals, biomass conversion, and fine chemicals can also have large disturbances and/or strict product specifications, which require effective handling of uncertainties with explicit characterization [22]. The intrinsic complexity of process dynamics, especially nonlinearity and uncertainty, has motivated the adoption of data-driven process control strategies.

B. Ideas and Classification of Data-Driven Process Control

The development of data-driven techniques for process control is typically based on the assumption of "*big data*" availability. Such big data are either collected and stored from the history of process operations or generated by a "digital twin" (i.e., a high-fidelity simulator that uses detailed and accurate models based on the underlying physics and

*This work was supported by NSF-CBET.

[1]Wentao Tang is currently with Surface Operations, Projects and Technology, Shell Global Solutions (U.S.) Inc., Houston, TX 77082, USA wentao.tang@shell.com

[2]Prodromos Daoutidis with the Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, MN 55455, USA daout001@umn.edu

chemistry) [23]. If sufficient amount of informative data have been obtained (more than what is enough to identify a linear model), it is possible to apply state-of-the-art machine learning methods to obtain more accurate nonlinear dynamic models and thereby design nonlinear model-based controllers. Hence, the first level of data-driven process control is *data-driven dynamic modeling*.

Controllers are used for operating processes. After a controller is designed and launched for online operations, its performance must be maintained through tasks in two different categories, which we refer to as controller *calibration* and *monitoring*. In calibration, the discrepancy between the behavior of the actual closed-loop system and the anticipated behavior is accounted for as an effect of exogenous disturbances [24] or uncertain parameters [25], while the process is assumed to be still under normal operating conditions. Such discrepancies are quantitatively captured and used for the correction of future control actions. In monitoring, internal factors (e.g., plant-model mismatch, ill controller tuning, and valve stiction) as well as external reasons (e.g., abnormal feed condition or equipment malfunctioning) for controller performance deterioration, called faults, need to be detected and diagnosed [26]. Preferably, the root cause for the faults can be determined, so that instructions for human intervention can be given in time. The calibration and monitoring of controllers rely on the data collected in real time during online operations. Instead of being "big data" methods, *data-driven performance maintenance* demands effective utilization of real-time online "small" data.

The aforementioned two types of data-driven process control are still based on dynamic models (in the form of differential equations describing the evolution of states). Specifically, data are primarily used to build, calibrate, or monitor the models. It can be argued though that models are not essentially necessary for control; all that should be needed is some *control-relevant information*, which can be much simpler than a complete model. By directly learning such information from offline and online data and thus avoiding system identification, data-driven methods promise a highly automated paradigm of future control technology that is truly centered at data. This type of data-driven process control, compared to the previous two, has been less investigated so far. We use the terminology of Fliess et al. [27] and call it *model-free* control.

The above different types of data-driven process control strategies are conceptualized in Fig. 1. The rest of the paper is organized as follows. In Section II, III, and IV, we give a tutorial yet concise introduction to the main representative approaches for the three categories of data-driven process control that we classified above. Subsequently, we discuss three crucial and fundamental future research directions in Section V. Conclusions are given in Section VI.

II. Data-Driven Dynamic Modeling

We consider a process system governed by continuous-time control-affine nonlinear dynamics:

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \;\; y(t) = h(x(t)), \qquad (1)$$

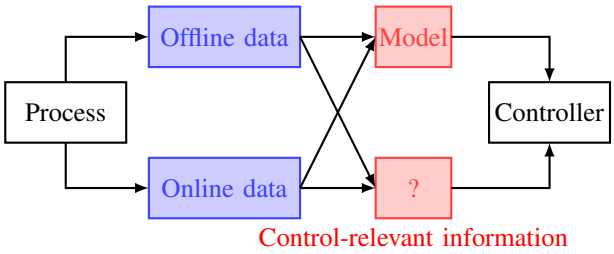

Fig. 1. Routes of data-driven process control.

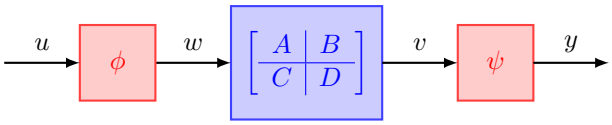

Fig. 2. Wiener-Hammerstein model.

where the vector of states is denoted by $x(t) \in X \subseteq \mathbb{R}^n$, inputs by $u(t) \in U \subseteq \mathbb{R}^m$, and outputs by $y(t) \in \mathbb{R}^p$. The functions $f : X \to \mathbb{R}^n$, $g : U \to \mathbb{R}^m$, and $h : X \to \mathbb{R}^p$ are smooth. The conditions for the nonlinear system (1) to be controllable and observable can be found in [28, Chapter 2], and we assume that these conditions are satisfied. The functions f, g, and h are assumed to be difficult to obtain through traditional system identification procedures [21] or parameter estimation on first-principles models [29]. In the presence of such complexity, data-driven techniques are needed to model the system.

It should remarked, however, that such nonlinear modeling may not always be needed for the whole system, but can be used for appending auxiliary nonlinear blocks after and before linear system models in a Wiener-Hammerstein model [30], [31]. That is, one can have u nonlinearly mapped onto $w \in \mathbb{R}^{m'}$ and $v \in \mathbb{R}^{p'}$ nonlinearly mapped onto y, so that a linear model exists between w and v:

$$\begin{aligned} \dot{x}(t) = Ax(t) + Bw(t), \;\; v(t) = Cx(t), \\ w(t) = \phi(u(t)), \;\; y(t) = \psi(v(t)). \end{aligned} \qquad (2)$$

Such a concept is illustrated in Fig. 2.

A. Sparse identification

For simplicity, let us first consider autonomous systems

$$\dot{x} = f(x) \qquad (3)$$

where the states can be measured. We assume that the function f can be expanded as a finite combination of multiple candidate basis functions $b_1, b_2, \ldots, b_K$, e.g., when f is a polynomial of degree $d \in \mathbb{N}$ and the basis functions consist of monomials of degree not exceeding d. That is,

$$\begin{aligned} f(x) &= \sum_{k=1}^{K} \theta_k b_k(x) \\ &= \begin{bmatrix} \theta_{11} & \cdots & \theta_{K1} \\ \vdots & \ddots & \vdots \\ \theta_{1n} & \cdots & \theta_{Kn} \end{bmatrix} \begin{bmatrix} b_1(x) \\ \vdots \\ b_K(x) \end{bmatrix} =: \Theta^\top b(x) \end{aligned} \qquad (4)$$

where $\theta_k \in \mathbb{R}^n$, $k = 1, \ldots, K$, i.e., $\Theta \in \mathbb{R}^{K\times n}$, and $b : X \to \mathbb{R}^K$. When observation pairs $\left(x^{(s)}, \dot{x}^{(s)}\right)$, $s = 1, \ldots, S$ are available (S is the number of data points), the parameters θ can be obtained through linear regression. In [32], it was proposed that such regression should promote the sparsity of θ, since the governing equation should be simple, i.e., f should not involve too many terms. For this purpose, an ℓ_1-penalty term is added for a LASSO formulation:

$$\min_{\Theta} \ \frac{1}{2}\|D - B\Theta\|^2 + \lambda\|\Theta\|_1. \tag{5}$$

In the above formulation,

$$B = \begin{bmatrix} b(x^{(1)})^\top \\ \vdots \\ b(x^{(S)})^\top \end{bmatrix}, \quad D = \begin{bmatrix} \dot{x}^{(1)\top} \\ \vdots \\ \dot{x}^{(S)\top} \end{bmatrix}, \tag{6}$$

and $\|\Theta\|_1 = \sum_{k=1}^{K}\sum_{i=1}^{n}|\theta_{ki}|$; $\lambda > 0$ is the regularization parameter.

The above approach based on basis expansion and sparse regression was termed *Sparse Identification of Nonlinear Dynamics* (SINDy) in [32]. Among the recent extensions, [33] proposed similar treatment for NARX model structures (for discrete-time systems), i.e.,

$$y(t) = \sum_{k=1}^{K} \theta_k f_k(y(t-1), \ldots, y(t-\delta_y), \\ u(t-1), \ldots, u(t-\delta_u)), \tag{7}$$

where $\delta_u, \delta_y \in \mathbb{N}$. Such NARX models overcome the limitation of state measurements, and useful heuristics to select the input–output terms f_k were discussed. [34] proposed to infer parameters θ through moving horizon optimization, which allows the use of online data.

B. Koopman operator theory

A different approach of identifying the governing equations of nonlinear dynamics is to find nonlinear transformations of the states whose dynamics is linear. Consider the autonomous system (3). Suppose that the initial condition is $x(0) = \xi$. Then after time t, the solution of (3) determines the state $x(t)$ dependent on ξ. We denote such a mapping (i.e., the *flow*) as $\mathcal{T}_t : \mathbb{R}^n \to \mathbb{R}^n$, $x(0) \mapsto x(t)$. Let us then consider any continuous complex vector-valued function of states $\varphi : X \to \mathbb{C}^N$, called observer function. The dynamics of such observer functions, is then represented by an operator defined on the function space, called the *Koopman operator*,

$$\mathcal{K}_t : \varphi \mapsto \mathcal{K}_t\varphi = \varphi \circ \mathcal{T}_t. \tag{8}$$

In other words, for any observer function φ, $\mathcal{K}_t\varphi(x(0)) = \varphi(x(T))$. This is illustrated in Fig. 3 with different choices of observables φ_1 and φ_2. It is not hard to verify that $\mathcal{K}_t$ is a linear operator on $C(X, \mathbb{C}^N)$ (the continuous function space). In this way, the Koopman operator approach achieves the *linearization of the system by lifting it onto an infinite-dimensional space*.

It is desirable that the infinite-dimensional Koopman operator $\mathcal{K}_t$ can be approximated as a finite-dimensional one, i.e.,

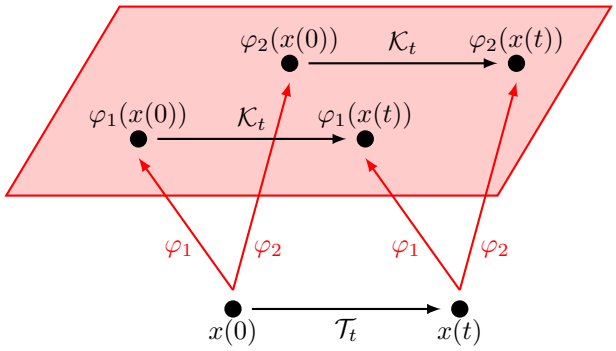

Fig. 3. Koopman operator of observer functions.

a matrix. Such an approximation based on input and output trajectory data is called *dynamic mode decomposition* (DMD) when the observer function comprises of the direct outputs y, or extended DMD, when nonlinear transformations of y are allowed [35] and has been often used for the analysis of fluid mechanical system [36]. Naturally, such a DMD procedure for dimension reduction is related to the truncation of data variations and can be realized through singular value decomposition (e.g., [37]).

Kaiser et al. [38] noted that, to ensure that the finite-dimensional approximation preserves the qualitative behavior, the observer functions need to be rationally chosen. Specifically, one needs to lift x by a $\varphi : X \to \mathbb{C}^m$ whose components span a subspace that is spanned by a number of eigenfunctions of $\mathcal{K}_t$. In this way, the truncation to a finite dimension ensures the closure of the eigen-subspace on which the dynamics is projected. An eigenfunction of $\mathcal{K}_t$ refers to a function $\omega : X \to \mathbb{C}$ such that there is a $\lambda \in \mathbb{C}$ satisfying

$$\mathcal{K}_t\omega = \omega \circ \mathcal{T}_t = e^{\lambda t}\omega, \ \ \forall t \in \mathbb{R}. \tag{9}$$

In the special case when the eigenvalue $\lambda = 0$, ω is said to be a conservable. It was proposed in [38] that such eigenfunctions can be approximately found through sparse regression.

For non-autonomous systems (1) with the control term $g(x)u$, it was assumed that if for $j = 1, \ldots, m$, each component of $(d\varphi(x)/dx)g_j(x)$ (g_j denotes the j-th column of g) can be expressed as an affine function of the components of $\varphi(x)$. Then with $z = \varphi(x)$, the system is converted to a bilinear form [39]:

$$\dot{z} = Az + \sum_{j=1}^{m}(p_j + Q_j z)u_j \tag{10}$$

for some $p_j \in \mathbb{R}^N$ and $Q_j \in \mathbb{R}^{N\times N}$, $j = 1, \ldots, m$. Such simple forms of dynamics can be analyzed with well-established theories such as optimal control [40].

C. Neural networks

In cases where the basis functions or the observers to be used in Koopman operators are difficult to know a priori, the nonlinearity in the model may need to be treated

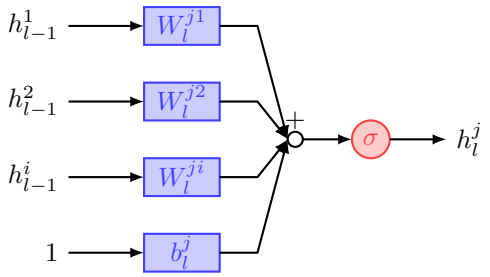

Fig. 4. A single neuron (perceptron) in the neural network.

totally as a black box function, which can be estimated by *universal approximators*. A typical type of representation is (multi-layer feedforward) neural networks. Denote the (vector of) inputs and outputs of the neural network as i and o, respectively, and the layer of hidden nodes as $L \in \mathbb{N}$. As illustrated in Fig. 4, for each layer $l = 1, \ldots, L$, the hidden states h_l are defined on the hidden states of the previous layer by an affine transformation fed into an element-wise nonlinear function. That is,

$$h_l = \sigma(W_l h_{l-1} + b_l), \tag{11}$$

in which W_l and b_l are matrix and vector of compatible dimensions, respectively, and the activation function σ is typically chosen to be $\tanh(\cdot)$ or the so-called rectified linear unit (ReLU, defined as $\mathrm{ReLU}(x) = x$ if $x \geq 0$ and $\mathrm{ReLU}(x) = 0$ otherwise). $h_0 = i$, and $h_L = o$.

Given input and output samples $\left(i^{(s)}, o^{(s)}\right)$, $s = 1, \ldots, S$, the neural networks can be trained to minimize the cost J, defined as mean squared errors:

$$J = \frac{1}{S} \sum_{s=1}^{S} \|o^{(s)} - \hat{h}_L^{(s)}\|^2, \tag{12}$$

with respect to the parameters $\theta = (\theta_0, \ldots, \theta_L)$, $\theta_l = (W_l, b_l)$. To seek a local minimum, a gradient descent algorithm can be used. To evaluate the gradient of J with respect to any b_l (analogously for W_l), one only needs to evaluate $\partial \hat{h}_L^{(s)} / \partial b_l$. According to the definition,

$$\frac{\partial \hat{h}_L^{(s)}}{\partial b_l} = \frac{d\sigma}{dx}(W_L \hat{h}_{L-1}^{(s)} + b_L) \cdot \begin{cases} W_L \frac{\partial \hat{h}_{L-1}^{(s)}}{\partial b_l}, & \text{if } l < L \\ I, & \text{if } l = L. \end{cases} \tag{13}$$

As such, the partial derivatives can be recursively computed by tracing back in the layers up to level l. This is called *back-propagation*, which is enabled by built-in automatic differentiation [41].

The introduction of neural network models in process control dates back to the early 1990s along with the establishment of the theoretic guarantee that they are universal approximators [42], [43]. In addition to being used as a black-box approximator for static nonlinear functions [44], *recurrent neural networks* (RNNs) can be used to encode the dynamics [45]. Specifically, in a single-layer continuous-time RNN [46], the evolution of each state x_i is described by

$$\dot{x}_i = -a_i x_i + \sum_{j=1}^{n} \left(\theta_{ij} \sigma(x_j) + b_{ij} u_j\right), \quad i = 1, \ldots, n \tag{14}$$

where $a_i > 0$, $\theta_{ij}, b_{ij} \in \mathbb{R}$, and σ is the nonlinear activation function as before. Motivated by the vast use of deep learning techniques in artificial intelligence, different approaches for multilayer RNN-based dynamic modeling using *long short-term memory* (LSTM) architectures have been studied in the recent process control literature [47]–[49].

D. Gaussian processes

Consider the approximation of a multivariable scalar-valued function $y = \phi(x) + \epsilon(x)$, where ϵ is a white noise:

$$\mathbb{E}[\epsilon(x)] = 0, \;\; \mathbb{E}[\epsilon(x)\epsilon(x')] = \sigma_\epsilon^2 \delta(x - x'), \tag{15}$$

in which $\delta(\cdot)$ is the Dirac's delta function. With a probabilistic point of view, the function ϕ is considered as a stationary Gaussian stochastic process [50]. That is, for any x, $\phi(x)$ is a normally distributed random variable; its prior distribution is given by $\phi(x) \sim \mathcal{N}(m(x), \sigma_\phi^2)$, and for any $x \neq x'$,

$$\mathbb{E}\left[(\phi(x) - m(x))\,(\phi(x') - m(x'))\right] = k(x, x'), \tag{16}$$

where $m(x) = 0$ without loss of generality, and $k(x, x')$ is a kernel function, e.g.,

$$k(x, x') = \sigma_\phi^2 \exp\left[-\frac{1}{2}(x - x')^\top R^{-1}(x - x')\right], \tag{17}$$

where R is a positive definite matrix.

Suppose that one has collected samples $(x^{(s)}, y^{(s)})$ for $s = 1, \ldots, S$. Consider any x. The joint prior distribution of the corresponding y and $y^{(s)}$ should be

$$\left(y, y^{(1)}, \ldots, y^{(S)}\right) \sim \mathcal{N}\Bigg(\left(m(x), m(x^{(1)}), \ldots, m(x^{(S)})\right), \begin{bmatrix} k(x,x) & k(x,x^{(1)}) & \cdots & k(x,x^{(S)}) \\ k(x^{(1)},x) & k(x^{(1)},x^{(1)}) & \cdots & k(x^{(1)},x^{(S)}) \\ \vdots & \vdots & \ddots & \vdots \\ k(x^{(S)},x) & k(x^{(S)},x^{(1)}) & \cdots & k(x^{(S)},x^{(S)}) \end{bmatrix} + \sigma_\epsilon^2 I\Bigg). \tag{18}$$

Let us denote $\bar{x} = (x^{(1)}, \ldots, x^{(S)})$, $\bar{y} = (y^{(1)}, \ldots, y^{(S)})$, and $K_{xx} = k(x, x)$, $K_{x\bar{x}} = [k(x, x^{(1)}), \ldots, k(x, x^{(S)})]$, $K_{\bar{x}x} = K_{x\bar{x}}^\top$, $K_{\bar{x}\bar{x}}$ alike, and $m(\bar{x}) = (m(x^{(1)}), \ldots, m(x^{(S)}))$, then

$$\begin{bmatrix} y \\ \bar{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(x) \\ m(\bar{x}) \end{bmatrix}, \begin{bmatrix} K_{xx} + \sigma_\epsilon^2 & K_{x\bar{x}} \\ K_{\bar{x}x} & K_{\bar{x}\bar{x}} + \sigma_\epsilon^2 I \end{bmatrix}\right). \tag{19}$$

With the observation obtained, the Bayes' rule generates a posterior Gaussian distribution:

$$\begin{aligned} y|\bar{y} \sim \mathcal{N}\big(&m(x) + K_{x\bar{x}}\left(K_{\bar{x}\bar{x}} + \sigma_\epsilon^2 I\right)^{-1}(\bar{y} - m(\bar{x})), \\ &K_{xx} + \sigma_\epsilon^2 - K_{x\bar{x}}\left(K_{\bar{x}\bar{x}} + \sigma_\epsilon^2 I\right)^{-1} K_{\bar{x}x}\big). \end{aligned} \tag{20}$$

The advantage of using a Gaussian process to model process dynamics is its stochastic nature, which allows the quantification of uncertainty alongside with the regression. Such uncertainties can be explicitly accounted for in the design of process control strategies, such as stochastic

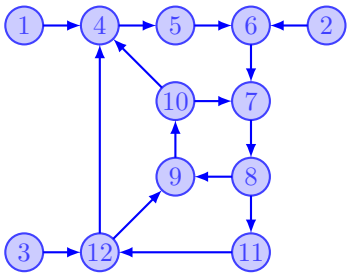

Fig. 5. Directed graph representation of system topology.

model predictive control [51], [52]. A key issue in Gaussian process-based MPC is to accurately capture the propagation of uncertainties during the prediction horizon.

E. Identification of system topology

For the modeling of large-scale systems, it is important to be aware that, instead of having a full model structure where all the outputs are dependent on most of the inputs, the underlying physics should make the interaction patterns between inputs and outputs *sparse*. The modeling should therefore be performed on a specified sparse structure (topology). When such a topology can not be completely determined according to domain knowledge of process systems, it needs to be identified from data. The problem is formulated as follows. Consider a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the each node $v \in \mathcal{V}$ represents a process variable and there exists a directed edge $(v, w) \in \mathcal{E}$ if variable v has a causal effect on w. With given trajectories of a part of the variables $\{v(\cdot)|v \in \mathcal{V}' \subseteq \mathcal{V}\}$, we are interested in reconstructing the topology of the graph, i.e., to find the set of hidden variables $\mathcal{V}'' = \mathcal{V} \backslash \mathcal{V}'$ and to reconstruct the edges $\mathcal{E}$. An illustration is given in Fig. 5. It should be expected that due to the existence of recycle streams, the topology of process systems should usually contain cycles.

Materassi and Salapaka [53] proposed that the locality of *Wiener filters* can be utilized for the reconstruction of unknown topology. They assumed that the (discrete-time, linear) dynamics of each node j can be expressed as

$$x_j(t) = e_j(t) + \sum_i H_{ji}(z) x_i(t) \tag{21}$$

where the transfer function H_{ji} is nonzero if there is an edge from node i to j, and the additive disturbance signals e_j, $j = 1, \ldots, n$ are mutually uncorrelated. No hidden node was considered. Wiener filters $\{W_{ji}(z)\}_{i \neq j}$ refers to the solution of the problem of finding transfer functions $W_{ji}(z)$ that are analytical on the unit circle to filter x_i, for all $i \neq j$, to minimize the signal approximation error:

$$\min_{W_{ji}} \left\| x_j - \sum_{i \neq j} W_{ji}(z) x_i \right\|^2 . \tag{22}$$

The solution is guaranteed to be unique. It was proved [53, Theorem 30] that $W_{ji}(z) \neq 0$ if and only if i is a parent node of j ($(i, j) \in \mathcal{E}$), a child node of j ($(j, i) \in \mathcal{E}$), or a co-parent of j (i.e., i and j have a common child). Moreover, if the Wiener filters are restricted to be *strictly causal*, then by replacing x_j with zx_j in (22), it is guaranteed that $W_{ji} \neq 0$ implies that i is a parent node of j [53, Theorem 35]. To prune the co-parent relations while avoiding strict causality, Talukdar et al. [54] proposed to exploit the phase information of $W_{ji}(z)$. In the presence of corrupted measurements, the corruption of the above Wiener filter-based topology identification procedure was studied [55].

When hidden nodes exist, under the assumption that the directed graph is a directed tree (polytree), Sepehr and Materassi [56] proposed a *pairwise-finite distance algorithm* (PFDA) that infers all the rooted trees in the polytree based on the cross-spectral density functions for all pairs of observed variables. Using graph-theoretic operations, the PFDA algorithm was extended in [57] to completely recover the directed tree, provided that the conditional independence between any two variables given any other variable (i.e., the third-order statistics) can be obtained. The exact identification of topology requires that each hidden node v in the latent graph satisfy the following conditions:

1) v has at least 2 children;
2) if v has only one parent, then its parent is not hidden;
3) if v has no parent and only 2 children, then both its children should have at least another parent.

Otherwise, the algorithm is capable of suggesting a simpler graph and thus plays the role of "Occam's razor".

For more general cases where $\mathcal{G}$ contains both hidden nodes and cycles, the recent work of Veedu et al. [58], based on the decomposition of inverse power spectral density matrix (IPSDM) into a sparse one and low-rank ones, distinguishes the parents, children, and co-parents of observed and hidden nodes. For nonlinear systems, nonlinear regression methods [59] and nonparametric, information-theoretic measures [60] can be used for topology construction.

III. Data-Driven Performance Maintenance

The data collected after the controller is launched for online operations can help to maintain the controller performance through various ways. In this section, we review four major different roles that online data can play to this end, namely to enable (i) modeling uncertainties including noise and disturbances, (ii) estimating parameters for adaptive and dual control, (iii) assessing performance, and (iv) analyzing faults. These topics are discussed in the following four subsections, respectively.

A. Noise and disturbance modeling

Consider a discrete-time linear system

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + B_d d(t) + Gw(t) \\ y(t) &= Cx(t) + Du(t) + D_d d(t) + v(t) \end{aligned} \tag{23}$$

where $d(t) \in \mathbb{R}^q$ represents the vector of disturbances, $w(t) \in \mathbb{R}^g$ and $v(t) \in \mathbb{R}^m$ are independent white noises with covariance matrices Q and R, respectively. For control, it is important to (i) handle the noises with a Kalman filter

designed based on the values of Q and R and (ii) estimate the disturbances $d(t)$ at any time t.

Now consider the case without disturbances ($d \equiv 0$) and the problem of estimating the unknown Q and R. Suppose that the model is known, and that input and output data $\{u(t), y(t)\}$ along with the state estimates $\{\hat{x}(t)\}$ generated along with a filter L:

$$\hat{x}(t+1) = A\hat{x}(t) + L\left(y(t+1) - A\hat{x}(t) - Bu(t)\right) \quad (24)$$

are available. The *autocovariance least-squares* (ALS) method of Odelson et al. [61] considers the innovations $\nu(t) = y(t) - A\hat{x}(t-1) - Bu(t)$ and their autocovariance matrices (which can be approximated by their sample estimations):

$$\Sigma_k = \mathbb{E}\left[\nu(t)\nu(t+k)^\top\right],\ k = 0, 1, \ldots, N-1 \quad (25)$$

up to a maximum delay $N-1$. Under suitable assumptions, it was proved that the stacked autocovariance matrix

$$\Upsilon_N = \begin{bmatrix} \Sigma_0 & \cdots & \Sigma_{N-1} \\ \vdots & \ddots & \vdots \\ \Sigma_{N-1}^\top & \cdots & \Sigma_0 \end{bmatrix} \quad (26)$$

satisfies the linear equality

$$\mathrm{vec}(\Upsilon_N) = \Theta_Q \mathrm{vec}(Q) + \Theta_R \mathrm{vec}(R), \quad (27)$$

where $\mathrm{vec}(\cdot)$ refers to the conversion of a matrix into a vector column-major order[1]. By performing least squares regression on (27), the noise covariance matrices Q and R can be obtained from measured input and output data. Such a regression can be weighted and regularized, subject to positive semidefiniteness constraints on Q and R, and is solvable through interior-point semidefinite programming algorithms [62], [63].

The second problem is the description of the disturbance dynamics. The primary motivation for disturbance modeling is to achieve offset-free control [64]–[66]. Muske and Badgwell [67] gave a model structure that allows both state disturbances and output disturbances as follows:

$$\begin{bmatrix} x(t+1) \\ d_x(t+1) \\ d_y(t+1) \end{bmatrix} = \begin{bmatrix} A & B_d & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} x(t) \\ d_x(t) \\ d_y(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} w(t) \\ \xi_x(t) \\ \xi_y(t) \end{bmatrix}$$
$$y(t) = \begin{bmatrix} C & 0 & D_d \end{bmatrix} x(t) + v(t). \quad (31)$$

Viewing (x, d_x, d_y) as the augmented states and (w, ξ_x, ξ_y, v) as the augmented noises, Kalman filters can be correspondingly designed to handle both disturbances and noises. Sun et al. [68] proposed that such disturbance models can be identified from closed-loop data by

1) performing a nonparametric regression to identify a high-order ARX model,
2) obtaining the extended observability and matrix $\bar{\Gamma}_f$ controllability matrix $\bar{L}_p$, and
3) fitting the state-space models into the above two matrices.

The readers are referred to the literature of subspace identification for detailed explanations [69].

B. Adaptive and dual control

For system (1), consider the case where f, g and h may not be known exactly; instead, there are some parameters $\theta \in \mathbb{R}^q$ that are uncertain and need to be determined during control. To estimate θ, *adaptation laws*, i.e., governing equations for the evolution of parameter estimates $\theta(t)$ need to be designed. Most commonly, an instantaneous cost $c(x(t), t)$ or integrated cost function $\int_0^t c(x(\tau), \theta(\tau), \tau)d\tau$ is defined as the objective, denoted as $J(t)$, and the adaption of $\theta(t)$ aims at decreasing $J(t)$ in the direction of gradient descent [70]. A comprehensive summary of the development of adaptive control methods since 1950s was given by the recent paper of Annaswamy and Fradkov [71].

Adaptive control methods for nonlinear systems in the form of (1), which is of special interest to process control applications, has been extensively studied in the 1990s [72], where input–output linearization was used and linear parameterization was typically assumed. For example, Marino and Tomei [73] considered the case where using a diffeomorphism $T : \mathbb{R}^n \to \mathbb{R}^n$, $x \mapsto \zeta = T(x)$, the dynamics (for simplicity, single-input-single-output) can be transformed into an *output feedback canonical form*:

$$\dot{\zeta} = A\zeta + b(\theta)\sigma(y)u + \psi_0(y) + \sum_{j=1}^{q} \psi_j(y)\theta_j$$
$$y = C\zeta \quad (32)$$

where

$$A = \begin{bmatrix} 0 & I_{n-1} \\ 0 & 0 \end{bmatrix},\ C = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}, \quad (33)$$

$\sigma : \mathbb{R} \to \mathbb{R}$, $b, \psi_j : \mathbb{R} \to \mathbb{R}^n$, $j = 0, 1, \ldots, m$ [73]. Kanellakopoulos et al. [74] assumed the conversion into a *parametric pure feedback* form:

$$\dot{\zeta}_i = \zeta_{i+1} + \theta^\top \gamma_i(\zeta_1, \ldots, \zeta_{i+1}),\ i = 1, \ldots, n-1$$
$$\dot{\zeta}_n = \gamma_0(\zeta) + \theta^\top \gamma_n(\zeta) + \left[\beta_0(x) + \theta^\top \beta(x)\right] u, \quad (34)$$

[1]The matrices Θ_Q and Θ_R are calculated as follows. Denote by $\otimes$ the Kronecker product operation, $M \oplus M := \mathrm{diag}(M, M)$ for any square matrix M, and hence $\oplus_{k=1}^N M = \mathrm{diag}(M, \ldots, M)$ where M appears in N diagonal blocks. Then denote $\bar{A} = A - ALC$,

$$O = \begin{bmatrix} C \\ C\bar{A} \\ \vdots \\ C\bar{A}^{N-1} \end{bmatrix},\ \Gamma = \begin{bmatrix} 0 & \cdots & 0 & 0 \\ C & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ C\bar{A}^{N-2} & \cdots & C & 0 \end{bmatrix}, \quad (28)$$

and $\Psi = -\Gamma\left(\oplus_{k=1}^N AL\right)$. Denote by I_n the n-th order unit matrix, and $J_{p,N} \in \{0,1\}^{(pN)^2 \times p^2}$ the matrix such that for any matrix M,

$$\mathrm{vec}\left(\oplus_{k=1}^N M\right) = J_{p,N}\mathrm{vec}(M). \quad (29)$$

Then,

$$\begin{aligned} \Theta_Q =& [(O \otimes O)(I_{n^2} - \bar{A} \otimes \bar{A})^{-1} + (\Gamma \otimes \Gamma)J_{n,N}](G \otimes G) \\ \Theta_R =& [(O \otimes O)(I_{n^2} - \bar{A} \otimes \bar{A})^{-1} + (\Gamma \otimes \Gamma)J_{n,N}](AL \otimes AL) \\ &+ [\Psi \oplus \Psi + I_{m^2N^2}]J_{p,N}. \end{aligned} \quad (30)$$

in which γ_i, $i = 0, 1, \ldots, n$, and β_0, β are smooth functions, and proposed a backstepping approach for adaptive control. More general nonlinearly parameterized systems were discussed in [75]–[77].

In the MPC context, Fukushima et al. [78] proposed the concept of *adaptive MPC* where the unknown parameter is updated and used to adjust the gain matrix of the auxiliary controller in the robust MPC, i.e., the controller inputs u is decomposed into two terms:

$$u(t) = -K(\theta)x(t) + \tilde{u}(t), \tag{35}$$

where $K(\theta)$ is the adaptive auxiliary controller and $\tilde{u}(t)$ is to be computed through optimization. In the formulations of Tanaskovic et al. [79], Lorenzen et al. [80], and Zhang and Shi [81], the online measurements are used to update the range of uncertain θ through falsification. The estimation of parameters, in the presence of nonlinearity, can be performed along with state observation through moving horizon estimation (MHE) [82]. Tang and Daoutidis [83] demonstrated that, when both the direct output measurements and output derivatives are used for the parameter estimation (called a *Lie-Sobolev approach*), the accuracy and control performance can be improved, even with structural model mismatch.

According to the philosophy of *dual control* by Feld'baum [84], when there exist uncertainties in the dynamics, the inputs u in the control should be responsible for both (i) regulating the closed-loop system, and (ii) exploring the dynamics by actively generating informative data, thus reducing the uncertainties. Heirung et al. [85] encoded such an idea in an MPC formulation. Specifically in dual MPC, one considers a finite impulse response (FIR) model with uncertain parameters $\theta \in \mathbb{R}^q$:

$$\varphi(t+1) = A\varphi(t) + Bu(t), \;\; y(t) = \theta^\top \varphi(t) + v(t) \tag{36}$$

in which the regressor $\varphi(t) \in \mathbb{R}^n$ plays the role of states. The parameters are updated through recursive least squares (Kalman filtering):

$$\theta(t) = \theta(t-1) + P(t)\varphi(t)\left[y(t) - \theta(t-1)^\top \varphi(t)\right] \tag{37}$$

where the regression matrix $P(t)$ is updated according to

$$\begin{aligned} &P(t+1) = P(t) - P(t)\varphi(t+1)\cdot \\ &\left[R + \varphi(t+1)^\top P(t)\varphi(t+1)\right]^{-1} \cdot \varphi(t+1)^\top P(t). \end{aligned} \tag{38}$$

Here R is the covariance matrix of white noise v. As such, in the prediction horizon, the variance of predicted outputs $\hat{y}(t+1)$ can be found as

$$\sigma_y^2(t+1) = \varphi(t+1)^\top P(t)\varphi(t+1) + R, \tag{39}$$

and can be used to analytically reformulate the chance constraints in a stochastic MPC formulation. Clearly, σ_y^2 decreases with more excitement in $\varphi(t)$, which on other hand deteriorates the regulation/tracking performance. The approach was extended from parametric uncertainties to structural uncertainties, for which online data-based model discrimination was used [86]. The readers are referred to the survey of Mesbah [87] for other explicit and implicit schemes for dual MPC control.

C. Control performance assessment

Automatically monitoring the performance of controllers in operations and diagnosing the reasons of performance degradation, based on the measured data of the process, is critical to the healthiness of industrial process control systems [26], [88]. First and foremost, this requires a quantitative measure to assess the controller's performance. A classical measure, given by Harris, is the ratio between the output variance of the minimum-variance controller (MVC) as a benchmark and the actual controller [89]. However, such benchmarking approaches require the availability of a hypothetical "ideal" controller.

For a data-based assessment, Yu and Qin [90] defined the *covariance performance index* as

$$I = \frac{\det \mathrm{cov}(y_{\mathrm{II}})}{\det \mathrm{cov}(y_{\mathrm{I}})}, \tag{40}$$

where y_{I} refers to the benchmark dataset, y_{II} is the monitored data, and cov stands for the sample covariance matrix. It is easy to verify that I is equal to the product of the eigenvalues of $M = [\mathrm{cov}(y_{\mathrm{I}})]^{-1}\mathrm{cov}(y_{\mathrm{II}})$ and is invariant to the scaling of individual components. In addition to an overall measure of control performance, the eigenspace decomposition of the matrix M reveals the contributions from different orthogonal directions in $\mathbb{R}^m$. Subsequently, in order to identify the responsible control loops and variables, the individual contributions (or the cosine values of the angles between the control loops and subspaces of M) can be computed [91]. Specifically, let

$$M = PLP^\top = \sum_{i=1}^{m} l_i p_i p_i^\top, \tag{41}$$

in which $p_i \in \mathbb{R}^m$, and λ_i be the asymptotic limit of the eigenvalue l_i as the number of samples approaches infinity. Statistical analysis [92] determines a confidence interval for λ_i with confidence $(1-\delta)$ according to[2]

$$1 - \zeta_{\delta/2}\beta_i \le l_i/\lambda_i \le 1 + \zeta_{\delta/2}\beta_i. \tag{44}$$

The assessment is then performed by testing the statistical significance of the comparison of λ_i to 1 according to the confidence region of λ_i specified by (44).

[2]Here, $\zeta_{\delta/2} \in \mathbb{R}$ is the point the cumulative density function of the standard normal distribution is evaluated $1 - \delta/2$. Let $s^2_{\mathrm{I}(i)}$ and $\sigma^2_{\mathrm{I}(i)}$ be the sample covariance and population variance of $z_{\mathrm{I}i} = p_i y_{\mathrm{I}}$, respectively, and $s^2_{\mathrm{II}(i)}$ and $\sigma^2_{\mathrm{II}(i)}$ analogously. Denote the auto-correlation coefficients of $z_{\mathrm{I}i}$ and $z_{\mathrm{II}i}$ at lag k as $\rho^k_{\mathrm{I}(i)}$ and $\rho^k_{\mathrm{II}(i)}$, respectively, and let

$$f_{\mathrm{I}(i)} = 1 + 2\sum_{k=1}^{n_{\mathrm{I}}} \left(1 - \frac{k}{m}\right)\left(\rho^k_{\mathrm{I}(i)}\right)^2, \tag{42}$$

where n_{I} is the number of samples for y_{I}; $f_{\mathrm{II}(i)}$ is defined alike. Then, β_i is calculated as

$$\gamma_i = \sqrt{2}\left(\frac{f_{\mathrm{I}(i)}}{n_{\mathrm{I}} - 1} + \frac{f_{\mathrm{II}(i)}}{n_{\mathrm{II}} - 1}\right)^{1/2}, \tag{43}$$

for all $i = 1, \ldots, m$.

For model-based controllers, checking the model quality through *plant-model mismatch detection* is deemed to be a more practical approach for control performance assessment [26]. Such a task can be performed by imposing open-loop exogenous input signals at the expense of perturbing the process from the equilibrium [93], [94]. To detect mismatch from closed-loop data, it is often assumed that a nominal disturbance model is known a priori, so that the controller performance can be benchmarked. Sun et al. [95] defined the model quality index as the ratio between averaged squares of model residuals (under an MVC benchmark) and the disturbance innovations of the monitored system. Other closed-loop approaches can be found in [96], [97]. In particular, Badwe et al. [98] proposed a method based on partial correlation analysis that is capable of locating the mismatch on single-input-single-output components of the model. Specifically, the method requires the availability of setpoint signals $r(t) \in \mathbb{R}^m$ that are uncorrelated with disturbances and uses the following steps.

- Correlate u with r and remove the uncorrelated part to obtain $\hat{u} = S_{ur}(z)r$.
- Decorrelate each $\hat{u}_j$, $j = 1, \ldots, m$ with the rest of the components u_i, $i \neq j$ to obtain ϵ_{u_i}.
- Decorrelate the j-th component of the error signal $e = y - G(z)u$ (G is the nominal model) with u_i, $i \neq j$ to obtain ϵ_{e_j}.
- Test the correlation between ϵ_{u_i} and ϵ_{e_j}. If the correlation is significant, then there is mismatch in $G_{ji}(z)$.

D. Fault detection and diagnosis

Discovering and analyzing faults in an early and accurate manner is of critical importance for the safety and maximal time of normal operations [99]. Since models for faults are often not available or reliable, fault detection and diagnosis typically uses data-driven methods [100]. The essence of fault diagnosis boils down to classification problems in statistics or machine learning, i.e., that of characterizing the distribution of fault-free data only (one-class classification, also known as anomaly/novelty detection), that of distinguishing faulty data with fault-free data, and that of classifying the measured data into one of multiple categories of faults. As a result, a plethora of statistical and machine learning-based methods have been developed (see [101]–[103] for reviews). A recent trend is the application of deep learning, which provides a useful toolbox that can handle nonlinearity, non-Gaussian distributions, temporal correlativity, multiscale characteristics, and mixture of labeled and unlabeled data [104]–[108].

We simply review two basic approaches here, namely principal component analysis (PCA) and partial least squares (PLS), to present the fundamental ideas underlying the vast literature on fault detection and diagnosis. Suppose that we have S measured data samples $x^{(s)} \in \mathbb{R}^n$, $s = 1, \ldots, S$ and want to characterize the distribution of the population. PCA assumes that the distribution is zero-mean and Gaussian, and first carries out the dimension reduction. Specifically, let $X = [x^{(1)}, \ldots, x^{(S)}]^\top \in \mathbb{R}^{S \times n}$, by singular value decomposition, one obtains

$$X = U\Sigma V^\top = \sum_{k=1}^{n} \sigma_i u_i v_i^\top, \tag{45}$$

where $U = [u_1, \ldots, u_S] \in \mathbb{R}^{S \times S}$, $V = [v_n, \ldots, v_n] \in \mathbb{R}^{n \times n}$ are orthogonal matrices, and $\sigma_1 \geq \sigma_2 \geq \ldots$. By truncating the terms with negligible singular values, one approximates X with the first $n_\mathrm{p} \ll n$ terms that capture a major portion of data variation:

$$X \approx TV^\top, \tag{46}$$

in which $T = [(t^{(1)})^\top; \ldots; (t^{(S)})^\top] := [\sigma_1 u_1, \ldots, \sigma_{n_\mathrm{p}} u_{n_\mathrm{p}}]$ is the matrix of *score vectors*, i.e., $V^\top x^{(s)} = t^{(s)}$, $s = 1, \ldots, S$. In this way, each sample of dimension n is reduced to n_p. The sample covariance of $\{t^{(s)}\}_{s=1}^S$ is thus $\Sigma_t = \frac{1}{S-1}\mathrm{diag}(\sigma_1^2, \ldots, \sigma_{n_\mathrm{p}}^2)$. Given any new data x, by calculating $t = V^\top x$ and subsequent statistics, one can then decide with a confidence level whether x is a faulty data. In PLS, it is further assumed that quality labels $y^{(s)} \in \mathbb{R}^p$ are available along with the measurements $x^{(s)}$. Decompositions are sought for X and Y, respectively:

$$X = TV_X^\top + E_X, \;\; Y = TV_Y^\top + E_Y, \tag{47}$$

so that the quality of any test data can be predicted.

Although statistical and machine learning approaches are useful for fault classification, it is desirable to understand the path of fault propagation and thus infer the root cause of the detected faults. The concept of *transfer entropy* proposed in Bauer et al. [109] quantifies the causality from one variable (x) to another (y) by the amount of information of future y contained in the past x. Consider discrete-time measurements $x(t)$ and $y(t)$. Denote by $\bar{x}(t) = (x(t), x(t-1), \ldots, x(t - \delta_x))$ and $\bar{y}(t) = (y(t), y(t-1), \ldots, y(t - \delta_y))$ as the history of x and y at time t. One can define the conditional probability according to the Bayes' rule:

$$p(y(t+h)|\bar{x}(t), \bar{y}(t)) = \frac{p(\bar{x}(t), \bar{y}(t)), y(t+h)}{p(\bar{x}(t), \bar{y}(t))}, \tag{48}$$

denoted as $p(y_h|\bar{x}, \bar{y})$ (considering (t) as a sampling action). Thus the information transferred from the history of x to the y after a delay of h is

$$\tau(y|x) = \int_{\mathbb{R}^{\delta_x + \delta_y + 1}} p(\bar{x}, \bar{y}, y_h) \ln \frac{p(y_h|\bar{x}, \bar{y})}{p(y_h|\bar{y})} d\bar{x} d\bar{y} dy_h \tag{49}$$

Then, the causality measure is defined as

$$\tau_{x \to y} = \tau(y|x) - \tau(x|y), \tag{50}$$

and the statistical significance of its evaluation based on samples can be tested against non-causal simulations. Different measures for variable causality were proposed in Yu and Rashid [110], where a dynamic Bayesian network (DBN) is used to capture the underlying interaction topology among variables. The approach was extended in [111], where kernel density estimation (KDE) is used to ease the computation of probability density functions. To discover such a DBN topology from data, the tools from *structure learning* can be used to optimize the graph structure with respect to a Bayesian information criterion (BIC) [112].

IV. Data-Driven Model-Free Control

In model-free control, we assume that the process dynamics is entirely a black box and we are not interested in recovering a dynamic model. To obtain a controller with satisfactory performance in this setting, different ideas have been proposed.

A. Training of model-free controllers

A natural idea of making controllers free from models is to *replace* model-based controllers with model-free ones, which essentially requires to use machine learning techniques to approximate a given function κ (the original model-based control law) with a new black-box data-based $\hat{\kappa}$ that does not contain any model information. This idea has been discussed in the context of *explicit MPC*, which stemmed from the multi-parametric programming approach of Bemporad et al. [113]. Specifically, for linear MPC, since the dependence of optimal control actions, as the solution to a quadratic programming problem, on the states should be a piecewise affine mapping, the repetitive optimization steps in MPC can be avoided with an a priori state space partitioning and an explicit look-up table. For nonlinear systems, such steps can be carried out approximately [114].

However, the exact conversion of implicit MPC laws into piecewise explicit mappings is computationally challenging and highly non-scalable. Parisini and Zoppoli [115] first proposed to approximate the MPC law as a neural network and proved closed-loop stability under this neural network controller on the basis of universal approximation property of neural networks. Chen et al. [116] proposed to train such networks in two phases that respectively decrease (i) primal infeasibility (i.e., the violation of operational constraints) and (ii) duality gap, which is related to the Karush-Kuhn-Tucker (KKT) conditions as a criterion of closeness to optimality. Recently, Kumar et al. [117] adopted a purposeful choice of the neural network architecture, which takes into setpoints $(x_{\text{sp}}, u_{\text{sp}})$ as inputs and uses structured parameterization:

$$\begin{aligned} h_0 &= (x, x_{\text{sp}}, u_{\text{sp}}, x_{\text{sp}}, x_{\text{sp}}, u_{\text{sp}}), \\ h_l &= \sigma\left(\begin{bmatrix} W_l & 0 \\ 0 & W_l \end{bmatrix} h_{l-1} + \begin{bmatrix} b_l \\ b_l \end{bmatrix}\right), \quad l = 1, \dots, L \\ u &= u_{\text{sp}} + \begin{bmatrix} W_{L+1} & -W_{L+1} \end{bmatrix} h_L. \end{aligned} \tag{51}$$

In this way, the neural network controller $\hat{\kappa}$ guarantees that

$$\hat{\kappa}(x_{\text{sp}} | x_{\text{sp}}, u_{\text{sp}}) = x_{\text{sp}}. \tag{52}$$

Also, the scalability improvement was enabled by generating only data that capture typical operation scenarios. The authors trained an industrial-scale MPC with $n = 252$, $m = 32$, and a prediction horizon of 140. The resulting explicit neural network controller computes 3 to 5 orders of magnitude faster, with performance loss $< 1\%$.

Apart from making a model-free controller imitate a model-based controller, another idea of training controllers, which has been studied for linear systems, is to directly optimize the controller parameters with respect to the desired closed-loop response and the input and output data. For example, in *virtual reference feedback tuning* (VRFT) [118], as illustrated in Fig. 6, one considers the tuning of controller $K(s|\theta)$ so that the desired closed-loop transfer function from the reference signals r to outputs y is close to $M(s)$. Given the input and output trajectories $u(\cdot)$ and $y(\cdot)$ on time interval $[0, T]$, the virtual reference $\hat{r}(\cdot)$ is first calculated such that $y = M(s)\hat{r}$, based on which the error $e = \hat{r} - y$ can be obtained. Then θ is given by

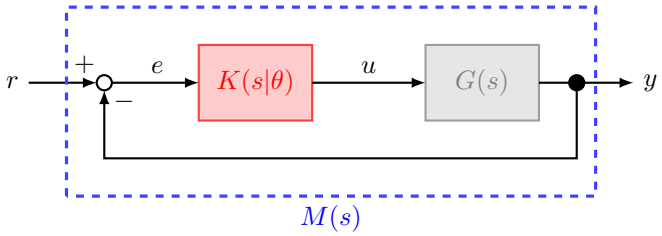

Fig. 6. Setup of IFT and VRFT.

$$\min_{\theta} J(\theta) = \frac{1}{2T} \int_0^T \left\| L(s)u(t) - L(s)K(s|\theta)e(t) \right\|^2 dt, \tag{53}$$

where $L(s)$ is a suitably chosen pre-filter. Apparently, the above problem does not explicitly involve the unknown dynamic model $G(s)$.

Prior to VRFT, the *iterative feedback tuning* (IFT) approach was proposed by Hjalmarsson et al. [119]. In IFT, the controller parameters θ are iteratively updated and the input and output trajectories are re-collected under the updated controller tuning. Denote by $u(t|\theta)$ and $y(t|\theta)$ the input and output data, and $\tilde{y}(t|\theta) = M(s)r(t) - y(t|\theta)$ the deviation of actual output signals away from the desired ones. Based on the definition of a minimization problem

$$\min_{\theta} \ J(\theta) = \frac{1}{2T} \int_0^T \left\| \tilde{y}(t|\theta) \right\|^2 dt, \tag{54}$$

the update of θ can be carried out through a gradient descent algorithm (i.e., $\theta \leftarrow \theta - \alpha \nabla J(\theta)$). The gradient $J(\theta)$ relies on the evaluation of $\partial y(t|\theta)/\partial\theta$, for which Hjalmarsson [120] designed a two-step experiment to derive a model-free unbiased estimation (i.e., the update algorithm is in fact stochastic gradient descent).

B. Learning of plant behaviors

In the case that we do not have a model-based controller at hand to train the model-free controller, it is still possible to design a controller without a complete dynamic model. The *dissipativity learning control* (DLC) framework was proposed in the previous research of the authors of this paper [121]–[123]. DLC consists of two major steps, namely the learning of the system's dissipativity properties and optimal controller synthesis. Dissipativity refers to the property that there exists a positive definite *storage* function $V(x)$ and an input- and output-dependent *supply rate* function $s(u, y)$ such that the dissipative inequality

$$\dot{V}(x) \leq s(u, y) \tag{55}$$

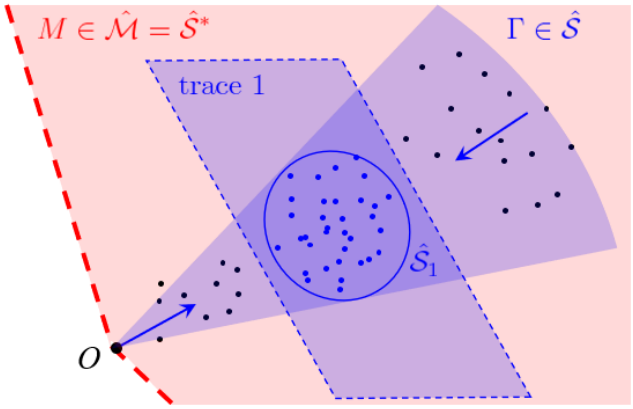

Fig. 7. Inference of the dissipativity set.

always holds. As proven by Hill and Moylan [124], the above condition can also be alternatively expressed as such that for any trajectory of any length T starting from the origin,

$$\int_0^T s\left(u(t), y(t)\right) dt \geq 0. \tag{56}$$

By analyzing the underlying thermodynamic laws, it was known that dissipativity is a common and anticipated property of process systems [125]–[127]. However, it is cumbersome and often challenging to obtain the dissipativity through such purely physical analysis for nonlinear systems in general.

Let us consider the supply rate function as a quadratic form of inputs and outputs

$$s = \begin{bmatrix} y^\top & u^\top \end{bmatrix} \begin{bmatrix} M_{yy} & M_{yu} \\ M_{uy} & M_{uu} \end{bmatrix} \begin{bmatrix} y \\ u \end{bmatrix} \tag{57}$$

so that function s can be parameterized by a symmetric matrix M. Substituting the above parameterization into (56), we have

$$\operatorname{trace}(M\Gamma) := \operatorname{trace}\left(M \int_0^T \begin{bmatrix} y \\ u \end{bmatrix} \begin{bmatrix} y^\top & u^\top \end{bmatrix} dt \right) \geq 0. \tag{58}$$

In the above formulation, the positive semidefinite matrix Γ is calculated based on the trajectory. We call the range of Γ for all the possible trajectories of the system as the *dual dissipativity set* (denoted as $\mathcal{S}$), and the range of valid M matrices as the *dissipativity set* (denoted as $\mathcal{M}$). Since both $M \in \mathcal{M}$ and $\Gamma \in \mathcal{S}$ are symmetric, the trace of their product can be seen as an inner product, i.e., $\operatorname{trace}(M\Gamma) =: \langle M, \Gamma \rangle$. Hence, the dissipativity set $\mathcal{M}$ is the *dual cone* of the dual dissipativity set:

$$\mathcal{M} = \mathcal{S}^* = \{M | \langle M, \Gamma \rangle \geq 0, \, \forall \Gamma \in \mathcal{S}\}. \tag{59}$$

Given trajectory samples $(u^{(s)}(t), y^{(s)}(t))$, $t \in [0, T^{(s)}]$, $s = 1, \ldots, S$, the corresponding dual dissipativity parameters $\Gamma^{(s)}$, $s = 1, \ldots, S$ can be calculated. Based on this dataset, the dissipativity set $\mathcal{S}$ is inferred through the following steps, as illustrated by Fig. 7.

- Since $\mathcal{S}$ is unbounded, scale $\Gamma^{(s)}$ so that their traces are equal to 1;
- Use principal component analysis (PCA) [123] or independent component analysis (ICA) [122] (ICA) to obtain an ellipsoidal or polyhedral range ($\hat{\mathcal{S}}_1$), respectively, of the samples, so that an estimation of $\mathcal{S}$ is $\hat{\mathcal{S}} = \{c\Gamma | c > 0, \Gamma \in \hat{\mathcal{S}}_1\}$, the cone of $\hat{\mathcal{S}}_1$;
- Characterize the dual cone of $\hat{\mathcal{S}}$, denoted as $\hat{\mathcal{M}}$, which is an estimation of $\mathcal{M}$. When $\hat{\mathcal{S}}_1$ is an ellipsoid or a polyhedron, $\hat{\mathcal{S}}$ is a second-order cone or a polyhedral cone, respectively.

Based on the inference of $\mathcal{M}$, the controller synthesis step seeks a controller $u = Ky$ so that for the closed-loop system, the resulting supply rate function

$$s(Ky, y) = y^\top \begin{bmatrix} I & K^\top \end{bmatrix} \begin{bmatrix} M_{yy} & M_{yu} \\ M_{uy} & M_{uu} \end{bmatrix} \begin{bmatrix} I \\ K \end{bmatrix} y \tag{60}$$

is a negative semidefinite quadratic form.

If the dynamics is considered to have exogenous disturbances d in addition to control inputs u, whose trajectories can also be measured, then the parameterization M has blocks associated with d as well. The controller synthesis can seek a controller K (within some a priori specified range $\mathcal{K}$) that minimizes the squared L_2-gain, β, from d to (u, y). Such an L_2-gain can be guaranteed if the following inequality is satisfied:

$$s(u, y, d) \leq \beta \|d\|^2 - \|u\|^2 - \|y\|^2. \tag{61}$$

The minimization of β thus reduces to the following semidefinite programming (SDP) problem:

$$\begin{aligned} \min_{\beta, K, M} \quad & \beta \\ \text{s.t.} \quad & \begin{bmatrix} I & K^\top & 0 \\ 0 & 0 & I \end{bmatrix} \left(+ \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & -\beta I \end{bmatrix} \right) \begin{bmatrix} I & 0 \\ K & 0 \\ 0 & I \end{bmatrix} \preceq 0, \\ & M \in \hat{\mathcal{M}}, \, K \in \mathcal{K}, \, \beta \geq 0. \end{aligned} \tag{62}$$

For linear systems, the learning of system behavior is more direct. According to Willems' fundamental lemma [128], the system behavior can be completely described by a single persistently exciting trajectory. We will discuss this further in Subsection V-C. Examples of such fundamental lemma-based model-free control include the "data-enabled predictive control" (DeePC) algorithm, where the model is replaced with a data-based expression in the optimal control problem [129]. An earlier idea, first proposed by Favoreel et al. [130], is to agglomerate the subspace identification, Kalman filter design, and linear quadratic regulator (LQR) design steps together so that the model is not explicitly involved in control. The idea was extended to predictive control by Kadali et al., called *subspace predictive control* [131].

C. Approximate dynamic programming and reinforcement learning

The above two types of model-free control methods use machine learning techniques either on the plant side or on

the controller side. It is possible to learn on both sides simultaneously. Consider the optimal (state-feedback) control problem of system (1). At any state $x \in X$, we look for a corresponding control input trajectory $\hat{u}(t) \in U, t \in [0, +\infty)$ that gives the minimum cost-to-go (with a discount rate $\rho \geq 0$):

$$\begin{aligned} \min_{\hat{u}(\cdot)} \; & V = \int_0^\infty e^{-\rho t} c\left(\hat{x}(t), \hat{u}(t)\right) dt \\ \text{s.t.} \; & \dot{\hat{x}}(t) = f(\hat{x}(t), \hat{u}(t)), \; t \in [0, +\infty) \\ & \hat{u}(t) \in U, \; t \in [0, +\infty) \\ & \hat{x}(0) = x. \end{aligned} \tag{63}$$

The initial value of the optimized input trajectory, $\hat{u}(0)$, as well as the optimal cost, are dependent on the initial state x; these two functions, are called the *optimal policy* (denoted by $u^*(x)$) and the *optimal value* (denoted by $V^*(x)$), respectively. They should satisfy the Hamilton-Jacobi-Bellman (HJB) equations:

$$\begin{aligned} u^*(x) &= \arg\min_u \left[\nabla V^*(x)^\top \left(f(x) + g(x)u\right) + c(x,u)\right], \\ \rho V^*(x) &= \nabla V^*(x)^\top \left(f(x) + g(x)u^*(x)\right) + c(x, u^*(x)). \end{aligned} \tag{64}$$

These partial differential equations generally do not permit analytical solutions and the numerical solution is difficult to scale up for high-dimensional systems [132]. Only when the state space is discrete (called a Markov decision process, or MDP, when the dynamics is stochastic), the Bellman equations become finite-dimensional and can be solved through *dynamic programming*. Yet, the discretization of the state space suffers from the "curse of dimensionality" (i.e., the number of grid points grows exponentially with the dimension).

To accommodate continuous state spaces and unknown dynamics, the idea of *approximate dynamic programming* (ADP) is to adopt function approximators and use samples of x and u to approximately solve the HJB equations [133]. Q-learning [134] is a typical approach to this end. The Q function, dependent on x and u, is defined as the infinimum of the remaining cost if the current states and inputs are x and u, respectively: [3]

$$\begin{aligned} Q(x,u) = \inf \; & \int_0^\infty e^{-\rho t} c(\hat{x}(t), \hat{u}(t)) dt \\ \text{s.t.} \; & \hat{x}(0) = x, \; \hat{u}(0) = u \\ & \hat{u}(\cdot) \text{ is Lipschitz continuous.} \end{aligned} \tag{65}$$

The Q should function should satisfy the following equations along the optimal control trajectory, for any $h > 0$:

$$Q(x(0), u(0)) = \int_0^h e^{-\rho t} c(x(t), u(t)) dt + Q(x(h), u(h)). \tag{66}$$

To learn the Q function in a parameterized form, e.g., a neural network, $Q(x, u|\theta_Q)$, one can iteratively optimize the control policy (with respect to Q), simulate the trajectories, update the left-hand side in (66) with the right-hand side of (66) under previous Q, and then improve the fit of θ. With Q function learned, the optimal policy and value functions can be recovered as

$$u^*(x) = \arg\min_u Q(x,u), \quad V^*(x) = \min_u Q(x,u). \tag{67}$$

For input-affine dynamics (which is often the case for process systems), (1), it may be possible to directly approximate u^* and V^*. Luo et al. [136] proved that given any trajectory $(u(t), x(t))$, $t \in [t_1, t_2]$, u^* and V^* should satisfy the following equation:

$$\begin{aligned} & V^*(x(t_1)) - V^*(x(t_2)) + \rho \int_{t_1}^{t_2} V^*(x) dt = \\ & \int_{t_1}^{t_2} \left[c\left(x, u^*(x)\right) + \left.\frac{\partial c}{\partial u}\right|_{(x, u^*(x))} \left(u - u^*(x)\right) \right] dt. \end{aligned} \tag{68}$$

Hence, the authors proposed to approximate u^* and V^* as neural networks (i.e., with an *actor-critic* architecture) and iteratively train them to fit the above equations with trajectory data. This idea was extended in Tang and Daoutidis [137] for the cases where the control inputs have bound constraints.

In the above-mentioned Q-learning, the estimation of $Q(x,u)$ is "off-policy", i.e., the update is based on a value of $u(h)$ that can be fully optimized. In the context of reinforcement learning (RL), the learning of the functions of interest can be performed in an "*on-policy*" manner, where the control policy and value functions can update themselves according to real-time, online data when the policy yet in exploration is actually executed. [138]. A representative approach for online RL is the "*state-action-reward-state-action*" (SARSA) algorithm [139] using an actor-critic neural network architecture, as illustrated in Fig. 8.

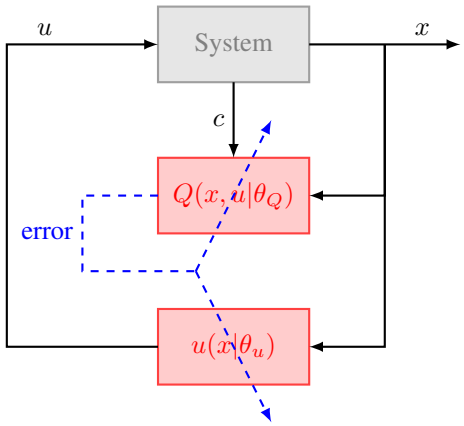

Fig. 8. Actor-critic architecture for reinforcement learning.

Having been successfully applied to a wide range of artificial intelligence and operations research problems [140], ADP and RL provide motivating approaches for model-free process control [141]. Yet, there are some common limitations that we should note.

- ADP and RL are based on optimal control theory and seek the solution of the HJB equations. The iterations in

[3] In fact, the discrete-time form is more frequently seen in the literature. Here we use the continuous-time form [135] for consistency with system (1).

the learning procedures are guaranteed to converge only if the function approximators are capable of capturing the true functions of interest. When the function is *mismatched*, the suboptimality of the resulting control performance can be difficult to characterize.

- For safety-critical process control, operational constraints must be imposed on the inputs and states. Recent research on *safe RL* has considered two different types of approaches – primal and primal-dual ones – to incorporate deterministic or chance constraints into RL and studied the corresponding complexity. The readers are referred to [142], [143] and the references therein for the pertinent works.
- The state measurements are involved. However, most of the states of process systems are usually not measurable, and for systems governed by partial differential equations (i.e., distributed parameter systems), the state space is infinite-dimensional. Since ADP and RL are not input–output approaches, the accompanying problem of designing a state observer with noise filtering will arise.
- The online implementation of RL relies on persistent perturbations of the process from the operating steady states. While it is known that such moves are necessary and that RL should have a tradeoff between "exploration" and "exploitation" [138], the optimal utilization of combined offline and online data (with respect to the process economics, the frequency of controller retuning, and the model degradation) needs to be better understood.

V. Future Directions

In the above three sections, we have introduced data-driven process control methods of three corresponding categories. Through the discussions in this section, we hope to suggest some important directions for future research in this field.

A. Scalability

Many process systems (e.g., refineries) are large-scale and high-dimensional, consisting of multiple interconnected process units or unit groups, and hence data-driven control algorithms must be scalable and structured. In other words, it is desirable to design such methods on the basis of the constituent *subsystems* of the process, so that

1) the complexities of learning and the resulting control do not grow drastically when the system dimension increases (i.e., the "curse of dimensionality" can be avoided),
2) the learning and control algorithms can be designed individually for the subsystems, with different tunings or different type of algorithms, to accommodate their heterogeneous dynamic characteristics, and
3) the malfunctioning, entrance or exit of subsystems can be tolerated.

In model-based control, such *distributed control* strategies can be designed by synthesizing a feedback law where (i) the feedback channels from the outputs to the inputs are restricted or regularized into the subsystem blocks [144], or (ii) the input–output responses are constrained to be localized [145]. Alternatively in the setting of MPC, distributed optimization algorithms (primal-dual algorithms based on operator splitting) [146] can be utilized to coordinate the solutions of the coupled receding horizon optimization problems of subsystems, with guaranteed convergence to the monolithic optima [147], [148]. Such principles will be useful for designing large-scale distributed data-driven controllers as well.

Also, the machine learning techniques applied on the basis of subsystems, which may be called as *distributed learning*, needs to be examined compared to the performance of monolithic centralized learning. For example, suppose that by learning the dissipativity of n interacting subsystems with inputs u_i, states x_i, and outputs y_i ($i = 1, \dots, n$), where some outputs are inputs to other subsystems), we have obtained supply rate functions s_i such that there exist storage functions V_i satisfying

$$\dot{V}_i(x_i) \leq s_i(u_i, y_i). \tag{69}$$

Then we can easily construct a storage function for the entire system $V(x) = \sum_i \alpha_i V_i(x_i)$ and a corresponding supply rate function $s(u, y) = \sum_i \alpha_i s_i(u_i, y_i)$ (where $\alpha_1, \dots, \alpha_n > 0$ can be arbitrarily chosen), so that

$$\dot{V}(x) \leq s(u, y). \tag{70}$$

However, the tightness of this directly implied inequality (although intuitive and frequently used, e.g., in [127], [149]) is unwarranted, and the resulting controller performance may be conservative. Therefore, it is desirable to coordinate the subsystem learners towards an optimal learning result for the monolithic system.

In order to determine a decomposition such that the coordination effort is minimal, it is desirable that the couplings between the resulting subsystems are weak compared to the interactions inside the subsystems, i.e., the interaction pattern in the system exhibits *modularity* with respect to the subsystems [150]. In the authors' previous research, the concept and tools of *community detection* in network science have been adopted to specify the subsystem allocations of inputs, states, and outputs [151], [152], and the advantages of such decompositions have been demonstrated in distributed MPC [153]. In the context of data-driven control (especially in the model-free setting), the effect of decompositions on the control and computational performance measures should be investigated along with the studies of distributed data-driven control for large-scale systems.

B. Physical interpretability

A concern often raised from the practitioners' stance is whether the information learned through data-driven techniques is in accordance with some physical ground truth. Indeed, it can be argued that physical interpretability may not always be necessary for data-driven techniques to work with desirable performance and that engineers' empirical dictations may be unreliable and error-prone. However, due to the criticality of safety in process systems, the implementation

of novel control technology tends to be difficult, and the capability of incorporating physical structures or constraints in data-driven techniques will be useful to facilitate acceptance.

Such requirements are typically met by adopting one or multiple among the following approaches.

1) Manual choice of features, namely the terms that contribute to the a priori physical laws governing the process dynamics in the dynamic modeling step. For example, chemical reaction kinetics is usually dependent on temperature according to the Arrhenius' law ($r \propto e^{-E/RT}$, where r is the rate of an elementary reaction, E is the activation energy, R is the ideal gas constant, and T is temperature). To apply Koopman operators to chemical reactors, [154] incorporated $\exp(-1/x)$ (where the state x is a scaled temperature) as a component of the observer function. For polynomial systems, it is intuitive that such observers should contain polynomial components (e.g., [35]).
2) Additional constraints on the ranges of the parameters learned from data. For example, in the dissipativity learning framework [122], it is physically required that for the supply rate function $s(u, y)$, when $y \to 0$, $s(u, 0) \geq 0$ must hold for any u, since the stabilized system must reach its minimum storage. Hence, when s is represented by a quadratic form $s(u, y) = [u^\top, y^\top]M[u; y]$, the corresponding block M_{uu} must be a positive semidefinite matrix. Because it is practically unrealistic to have infinitesimal trajectories around the origin, this physical constraint may not be reflected by the learning result, and therefore needs to be imposed a posteriori.
3) Identification or learning in function forms whose structures are already partly fixed according to simplified physical models, i.e., in (pretended) gray boxes. Such simple models exist for typical building blocks of process units, e.g., reaction rate laws approximated as those for elementary reactions and phase equilibrium equations approximated as those for ideal mixtures. This has been extensively used in the context of MPC combined with parameter estimation schemes (e.g., [155], [156]).
4) Regularization to promote sparsity, as an "Occam's razor" to enforce the simplicity of physical laws. This was mentioned in Section II.

The choice of specific approaches to incorporate physical first principles, however, is dependent on the corresponding data-driven techniques used and usually lacks comparison and clear guidelines. It is suggested that for the development of data-driven process control strategies, different extents of blending physical constraints or structures into learning should be allowed, so that it can be practical to optimize such blending through testing on actual processes or high-fidelity simulators.

C. Data-information relation

Key to the system identification in model-based control and learning of control-relevant information in model-free control is the conditions under which the trajectory samples "encode" the dynamics. Hence, the development of data-driven process control methods can not avoid answering the questions of "how much data are needed" and "what properties the data should have". This requires fundamental understanding of the relation between data and information, which is so far established mainly for linear systems, where a condition is given by the so-called *fundamental lemma* of Willems et al. [128].

Specifically, suppose that we have a single input and output trajectory $(u(t), y(t))$, $t = 0, 1, \ldots, N-1$ for a discrete, time-invariant linear system. The input signal (which is m-dimensional) is called persistently exciting of order r if the Hankel matrix

$$H_r(u) := \begin{bmatrix} u(0) & u(1) & \cdots & u(N-r) \\ u(1) & u(2) & \cdots & u(N-r+1) \\ \vdots & \vdots & \ddots & \vdots \\ u(r-1) & u(r) & \cdots & u(N-1) \end{bmatrix} \tag{71}$$

is of rank mr. It was proved in [128] that for any other $L \in \mathbb{N}$, under the assumption that the trajectory $(u(t), y(t))$ is persistently exciting of order $r = n + L$, any hypothetical trajectory $(\bar{u}(t), \bar{y}(t))$, $t = 0, 1, \ldots, L-1$ of length L is a possible trajectory of the system if there exists $\alpha \in \mathbb{R}^{N-L+1}$ such that

$$\begin{bmatrix} H_L(u) \\ H_L(y) \end{bmatrix} \alpha = \begin{bmatrix} \bar{u} \\ \bar{y} \end{bmatrix}. \tag{72}$$

In this way, all trajectories of the system, although not observed, can be parameterized with the available persistently exciting trajectory, and hence it is possible to learn the system's properties, e.g., dissipativity and integral quadratic constraints (IQCs), based on the data [157]. The strategies to optimize the trajectory to sample for effectively learning of such properties were also discussed in [158]. It is also natural to perform linear controller synthesis based on this parameterization [159].

For nonlinear systems, it should be anticipated that Willems' parameterization of trajectories (72) will no longer be achievable. Discussions about the fundamental data-information relation are scarce. Markovsky [160] proposed to embed generalized forms of bilinear systems into linear ones in order to apply the fundamental lemma. In the context of Koopman operators, Arbabi and Mezić [37] used ergodicity as an underlying condition to prove the asymptotic convergence of dynamic mode decomposition to the true Koopman operator within the minimal invariant set as the dimension of observer approaches infinity. However, the condition of ergodicity is restrictive and difficult to verify. It seems fair to claim that so far, the requirements on data for data-driven control of nonlinear systems remain an open question.

VI. Conclusions

In this paper, we have discussed the motivations for data-driven process control and gave an overview of the representative, pertinent approaches. These approaches are classified into three categories – data-driven dynamic modeling, performance maintenance, and model-free control –

according to the different roles and contributions of data-driven techniques in control. Three major issues – scaling up to high-dimensional systems, incorporating physical interpretation into learning, and characterizing the fundamental relations between data and control-relevant information – are identified as the most imperative aspects for defining future directions. We believe that the field of data-driven process control is still at an incipient stage of development and needs to be further explored before the research in this field can be turned into practically implemented technology.

REFERENCES

[1] Z.-S. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Inform. Sci.*, vol. 235, pp. 3–35, 2013.

[2] S. Yin and O. Kaynak, "Big data for modern industry: Challenges and trends [point of view]," *Proc. IEEE*, vol. 103, no. 2, pp. 143–146, 2015.

[3] B. A. Ogunnaike and W. H. Ray, *Process dynamics, modeling, and control*. Oxford University Press, 1994.

[4] V. Venkatasubramanian, "Drowning in data: Informatics and modeling challenges in a data-rich networked world," *AIChE J.*, vol. 55, no. 1, pp. 2–8, 2009.

[5] S. J. Qin and L. H. Chiang, "Advances and opportunities in machine learning for process data analytics," *Comput. Chem. Eng.*, vol. 126, pp. 465–473, 2019.

[6] C. D. Siebenthal and R. Aris, "Studies in optimization – VI. the application of Pontryagin's methods to the control of a stirred reactor," *Chem. Eng. Sci.*, vol. 19, no. 10, pp. 729–746, 1964.

[7] L. A. Gould, L. B. Evans, and H. Kurihara, "Optimal control of fluid catalytic cracking processes," *Automatica*, vol. 6, no. 5, pp. 695–703, 1970.

[8] G. P. Pollard and R. W. H. Sargent, "Offline computation of optimum controls for a plate distillation column," *Automatica*, vol. 6, no. 1, pp. 59–76, 1970.

[9] D. M. Prett and M. Morari, *The Shell process control workshop*. Butterworth, 1987.

[10] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Eng. Pract.*, vol. 11, no. 7, pp. 733–764, 2003.

[11] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model predictive control: Theory, computation, and design*, 2nd ed. Nob Hill Publishing, 2020.

[12] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[13] L. T. Biegler, "Advances in nonlinear programming concepts for process control," *J. Process Control*, vol. 8, no. 5-6, pp. 301–311, 1998.

[14] R. Amrit, J. B. Rawlings, and L. T. Biegler, "Optimizing process economics online using model predictive control," *Comput. Chem. Engineering*, vol. 58, pp. 334–343, 2013.

[15] J. Liu, D. Muñoz de la Peña, and P. D. Christofides, "Distributed model predictive control of nonlinear process systems," *AIChE J.*, vol. 55, no. 5, pp. 1171–1184, 2009.

[16] I. J. Wolf and W. Marquardt, "Fast NMPC schemes for regulatory and economic NMPC – A review," *J. Process Control*, vol. 44, pp. 162–183, 2016.

[17] M. Morari and E. Zafiriou, *Robust process control*. Prentice Hall, 1989.

[18] P. Daoutidis, M. Soroush, and C. Kravaris, "Feedforward/feedback control of multivariable nonlinear processes," *AIChE J.*, vol. 36, no. 10, pp. 1471–1484, 1990.

[19] J. H. Lee, M. Morari, and C. E. Garcia, "State-space interpretation of model predictive control," *Automatica*, vol. 30, no. 4, pp. 707–717, 1994.

[20] M. Soroush and C. Kravaris, "MPC formulation of GLC," *AIChE J.*, vol. 42, no. 8, pp. 2377–2381, 1996.

[21] L. Ljung, *System identification: Theory for the user*, 2nd ed. Prentice Hall, 1999.

[22] Q. Su, S. Ganesh, M. Moreno, Y. Bommireddy, M. Gonzalez, G. V. Reklaitis, and Z. K. Nagy, "A perspective on Quality-by-Control (QbC) in pharmaceutical continuous manufacturing," *Comput. Chem. Eng.*, vol. 125, pp. 216–231, 2019.

[23] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-art," *IEEE Trans. Ind. Inform.*, vol. 15, no. 4, pp. 2405–2415, 2018.

[24] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[25] D. E. Seborg, T. F. Edgar, and S. L. Shah, "Adaptive control strategies for process control: a survey," *AIChE J.*, vol. 32, no. 6, pp. 881–913, 1986.

[26] X. Gao, F. Yang, C. Shang, and D. Huang, "A review of control loop monitoring and diagnosis: Prospects of controller maintenance in big data era," *Chin. J. Chem. Eng.*, vol. 24, no. 8, pp. 952–962, 2016.

[27] M. Fliess and C. Join, "Model-free control," *Int. J. Control*, vol. 86, no. 12, pp. 2228–2252, 2013.

[28] A. Isidori, *Nonlinear control systems*. Springer, 1995.

[29] C. Kravaris, J. Hahn, and Y. Chu, "Advances and selected recent developments in state and parameter estimation," *Comput. Chem. Eng.*, vol. 51, pp. 111–123, 2013.

[30] H. Boutayeb and M. Darouach, "Recursive identification method for MISO Wiener-Hammerstein model," *IEEE Trans. Autom. Control*, vol. 40, no. 2, pp. 287–291, 1995.

[31] Y. Zhu, "Estimation of an N-L-N Hammerstein-Wiener model," *Automatica*, vol. 38, no. 9, pp. 1607–1614, 2002.

[32] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 113, no. 15, pp. 3932–3937, 2016.

[33] W. Sun and R. D. Braatz, "ALVEN: Algebraic learning via elastic net for static and dynamic nonlinear model identification," *Comput. Chem. Eng.*, vol. 143, p. 107103, 2020.

[34] F. Lejarza and M. Baldea, "Discovering governing equations via moving horizon learning: The case of reacting systems," *AIChE J.*, p. e17567, 2021.

[35] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition," *J. Nonlin. Sci.*, vol. 25, no. 6, pp. 1307–1346, 2015.

[36] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *J. Fluid Mech.*, vol. 656, pp. 5–28, 2010.

[37] H. Arbabi and I. Mezić, "Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator," *SIAM J. Appl. Dyn. Syst.*, vol. 16, no. 4, pp. 2096–2126, 2017.

[38] E. Kaiser, J. N. Kutz, and S. L. Brunton, "Data-driven discovery of Koopman eigenfunctions for control," *Mach. Learn. Sci. Technol.*, vol. 2, no. 3, p. 035023, 2021.

[39] D. Bruder, X. Fu, and R. Vasudevan, "Advantages of bilinear Koopman realizations for the modeling and control of systems with unknown dynamics," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4369–4376, 2021.

[40] M. Korda and I. Mezić, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149–160, 2018.

[41] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: A survey," *J. Mach. Learn. Res.*, vol. 18, pp. 1–43, 2018.

[42] N. Bhat and T. J. McAvoy, "Use of neural nets for dynamic modeling and control of chemical process systems," *Comput. Chem. Eng.*, vol. 14, no. 4-5, pp. 573–582, 1990.

[43] J. C. Hoskins and D. M. Himmelblau, "Process control via artificial neural networks and reinforcement learning," *Comput. Chem. Eng.*, vol. 16, no. 4, pp. 241–251, 1992.

[44] H. T. Su and T. J. McAvoy, "Integration of multilayer perceptron networks and linear dynamic models: A Hammerstein modeling approach," *Ind. Eng. Chem. Res.*, vol. 32, no. 9, pp. 1927–1936, 1993.

[45] R. K. Al Seyab and Y. Cao, "Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation," *J. Process Control*, vol. 18, no. 6, pp. 568–581, 2008.

[46] Z. Wu, A. Tran, D. Rincon, and P. D. Christofides, "Machine

learning-based predictive control of nonlinear processes. Part I: Theory," *AIChE J.*, vol. 65, no. 11, p. e16729, 2019.

[47] S. S. P. Kumar, A. Tulsyan, B. Gopaluni, and P. Loewen, "A deep learning architecture for predictive control," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 512–517, 2018.

[48] M. J. Ellis and V. Chinde, "An encoder–decoder LSTM-based EMPC framework applied to a building HVAC system," *Chem. Eng. Res. Des.*, vol. 160, pp. 508–520, 2020.

[49] E. Esche, J. Weigert, G. B. Rihm, J. Göbel, and J.-U. Repke, "Architectures for neural networks as surrogates for dynamic systems in chemical engineering," *Chem. Eng. Res. Des.*, vol. 177, pp. 184–199, 2022.

[50] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer school on machine learning*. Springer, 2003, pp. 63–71.

[51] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using Gaussian process regression," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 6, pp. 2736–2743, 2019.

[52] E. Bradford, L. Imsland, D. Zhang, and E. A. del Rio Chanona, "Stochastic data-driven model predictive control using Gaussian processes," *Comput. Chem. Eng.*, vol. 139, p. 106844, 2020.

[53] D. Materassi and M. V. Salapaka, "On the problem of reconstructing an unknown topology via locality properties of the Wiener filter," *IEEE Trans. Autom. Control*, vol. 57, no. 7, pp. 1765–1777, 2012.

[54] S. Talukdar, D. Deka, H. Doddi, D. Materassi, M. Chertkov, and M. V. Salapaka, "Physics informed topology learning in networks of linear dynamical systems," *Automatica*, vol. 112, p. 108705, 2020.

[55] V. R. Subramanian, A. Lamperski, and M. V. Salapaka, "Network structure identification from corrupt data streams," *IEEE Trans. Autom. Control*, vol. 66, no. 11, pp. 5314–5325, 2020.

[56] F. Sepehr and D. Materassi, "Blind learning of tree network topologies in the presence of hidden nodes," *IEEE Trans. Autom. Control*, vol. 65, no. 3, pp. 1014–1028, 2019.

[57] ——, "An algorithm to learn polytree networks with hidden nodes," in *Proc. 33rd Conf. Adv. Neur. Inform. Process. Syst. (NeurIPS 2019)*, 2019.

[58] M. Veedu, D. Harish, and M. V. Salapaka, "Topology learning of linear dynamical systems with latent nodes using matrix decomposition," *IEEE Trans. Autom. Control*, 2021, in press, doi:10.1109/TAC.2021.3124979.

[59] G. B. Giannakis, Y. Shen, and G. V. Karanikolas, "Topology identification and learning over graphs: Accounting for nonlinearities and dynamics," *Proc. IEEE*, vol. 106, no. 5, pp. 787–807, 2018.

[60] C. J. Quinn, N. Kiyavash, and T. P. Coleman, "Directed information graphs," *IEEE Trans. Inform. Theor.*, vol. 61, no. 12, pp. 6887–6909, 2015.

[61] B. J. Odelson, M. R. Rajamani, and J. B. Rawlings, "A new autocovariance least-squares method for estimating noise covariances," *Automatica*, vol. 42, no. 2, pp. 303–308, 2006.

[62] B. M. Åkesson, J. B. Jørgensen, N. K. Poulsen, and S. B. Jørgensen, "A generalized autocovariance least-squares method for Kalman filter tuning," *J. Process Control*, vol. 18, no. 7-8, pp. 769–779, 2008.

[63] M. R. Rajamani and J. B. Rawlings, "Estimation of the disturbance structure from data using semidefinite programming and optimal weighting," *Automatica*, vol. 45, no. 1, pp. 142–148, 2009.

[64] G. Pannocchia and J. B. Rawlings, "Disturbance models for offset-free model-predictive control," *AIChE J.*, vol. 49, no. 2, pp. 426–437, 2003.

[65] G. Pannocchia and A. Bemporad, "Combined design of disturbance model and observer for offset-free model predictive control," *IEEE Trans. Autom. Control*, vol. 52, no. 6, pp. 1048–1053, 2007.

[66] M. Morari and U. Maeder, "Nonlinear offset-free model predictive control," *Automatica*, vol. 48, no. 9, pp. 2059–2067, 2012.

[67] K. R. Muske and T. A. Badgwell, "Disturbance modeling for offset-free linear model predictive control," *J. Process Control*, vol. 12, no. 5, pp. 617–632, 2002.

[68] Z. Sun, Y. Zhao, and S. J. Qin, "Improving industrial MPC performance with data-driven disturbance modeling," in *Proc. Joint 50th IEEE Conf. Decis. Control, Eur. Control Conf. (CDC-ECC)*. IEEE, 2011, pp. 1922–1927.

[69] T. Katayama, *Subspace methods for system identification*. Springer, 2005, vol. 1.

[70] P. Ioannou and B. Fidan, *Adaptive control tutorial*. SIAM, 2006.

[71] A. M. Annaswamy and A. L. Fradkov, "A historical perspective of adaptive control and learning," *Annu. Rev. Control*, vol. 52, pp. 18–41, 2021.

[72] M. V. Druzhinina, V. O. Nikiforov, and A. L. Fradkov, "Methods of nonlinear adaptive control with respect to output," *Autom. Remote Control*, vol. 57, no. 2, pp. 153–176, 1996.

[73] R. Marino and P. Tomei, "Global adaptive output-feedback control of nonlinear systems. I. linear parameterization," *IEEE Trans. Autom. Control*, vol. 38, no. 1, pp. 17–32, 1993.

[74] I. Kanellakopoulos, P. V. Kokotović, and A. S. Morse, "Systematic design of adaptive controllers for feedback linearizable systems," *IEEE Trans. Autom. Control*, vol. 36, no. 11, pp. 1241–1253, 1991.

[75] W. Lin and C. Qian, "Adaptive control of nonlinearly parameterized systems: the smooth feedback case," *IEEE Trans. Autom. Control*, vol. 47, no. 8, pp. 1249–1266, 2002.

[76] M. Farza, M. M'Saad, T. Maatoug, and M. Kamoun, "Adaptive observers for nonlinearly parameterized class of nonlinear systems," *Automatica*, vol. 45, no. 10, pp. 2292–2299, 2009.

[77] I. Y. Tyukin, E. Steur, H. Nijmeijer, and C. Van Leeuwen, "Adaptive observers and parameter estimation for a class of systems nonlinear in the parameters," *Automatica*, vol. 49, no. 8, pp. 2409–2423, 2013.

[78] H. Fukushima, T.-H. Kim, and T. Sugie, "Adaptive model predictive control for a class of constrained linear systems based on the comparison model," *Automatica*, vol. 43, no. 2, pp. 301–308, 2007.

[79] M. Tanaskovic, L. Fagiano, R. Smith, and M. Morari, "Adaptive receding horizon control for constrained MIMO systems," *Automatica*, vol. 50, no. 12, pp. 3019–3029, 2014.

[80] M. Lorenzen, M. Cannon, and F. Allgöwer, "Robust MPC with recursive model update," *Automatica*, vol. 103, pp. 461–471, 2019.

[81] K. Zhang and Y. Shi, "Adaptive model predictive control for a class of constrained linear systems with parametric uncertainties," *Automatica*, vol. 117, p. 108974, 2020.

[82] P. Kühl, M. Diehl, T. Kraus, J. P. Schlöder, and H. G. Bock, "A real-time algorithm for moving horizon state and parameter estimation," *Comput. Chem. Eng.*, vol. 35, no. 1, pp. 71–83, 2011.

[83] W. Tang and P. Daoutidis, "Nonlinear state and parameter estimation using derivative information: A Lie-Sobolev approach," *Comput. Chem. Eng.*, vol. 151, p. 107369, 2021.

[84] N. M. Filatov and H. Unbehauen, *Adaptive dual control: Theory and applications*. Springer Science & Business Media, 2004, vol. 302.

[85] T. A. N. Heirung, B. E. Ydstie, and B. Foss, "Dual adaptive model predictive control," *Automatica*, vol. 80, pp. 340–348, 2017.

[86] T. A. N. Heirung, T. L. Santos, and A. Mesbah, "Model predictive control with active learning for stochastic systems with structural model uncertainty: Online model discrimination," *Comput. Chem. Eng.*, vol. 128, pp. 128–140, 2019.

[87] A. Mesbah, "Stochastic model predictive control with active uncertainty learning: A survey on dual control," *Annu. Rev. Control*, vol. 45, pp. 107–117, 2018.

[88] M. Jelali, "An overview of control performance assessment technology and industrial applications," *Control Eng. Pract.*, vol. 14, no. 5, pp. 441–466, 2006.

[89] T. J. Harris, "Assessment of control loop performance," *Can. J. Chem. Eng.*, vol. 67, no. 5, pp. 856–861, 1989.

[90] J. Yu and S. J. Qin, "Statistical MIMO controller performance monitoring. Part I: Data-driven covariance benchmark," *J. Process Control*, vol. 18, no. 3–4, pp. 277–296, 2008.

[91] ——, "Statistical MIMO controller performance monitoring. Part II: Performance diagnosis," *J. Process Control*, vol. 18, no. 3-4, pp. 297–319, 2008.

[92] L. Desborough and T. Harris, "Performance assessment measures for univariate feedback control," *Can. J. Chem. Eng.*, vol. 70, no. 6, pp. 1186–1197, 1992.

[93] G. Ji, K. Zhang, and Y. Zhu, "A method of MPC model error detection," *J. Process Control*, vol. 22, no. 3, pp. 635–642, 2012.

[94] S. Kaw, A. K. Tangirala, and A. Karimi, "Improved methodology and set-point design for diagnosis of model-plant mismatch in control loops using plant-model ratio," *J. Process Control*, vol. 24, no. 11, pp. 1720–1732, 2014.

[95] Z. Sun, S. J. Qin, A. Singhal, and L. Megan, "Performance monitoring of model-predictive controllers via model residual assessment," *J. Process Control*, vol. 23, no. 4, pp. 473–482, 2013.

[96] Q. Lu, M. G. Forbes, P. D. Loewen, J. U. Backström, G. A. Dumont, and R. B. Gopaluni, "Support vector machine approach for model-plant mismatch detection," *Comput. Chem. Eng.*, vol. 133, p. 106660, 2020.

[97] D. Ling, Y. Zheng, H. Zhang, W. Yang, and B. Tao, "Detection of model-plant mismatch in closed-loop control system," *J. Process Control*, vol. 57, pp. 66–79, 2017.

[98] A. S. Badwe, R. D. Gudi, R. S. Patwardhan, S. L. Shah, and S. C. Patwardhan, "Detection of model-plant mismatch in MPC applications," *J. Process Control*, vol. 19, no. 8, pp. 1305–1313, 2009.

[99] L. H. Chiang, E. L. Russell, and R. D. Braatz, *Fault detection and diagnosis in industrial systems*. Springer, 2000.

[100] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin, "A review of process fault detection and diagnosis: Part III: Process history based methods," *Comput. Chem. Eng.*, vol. 27, no. 3, pp. 327–346, 2003.

[101] S. J. Qin, "Survey on data-driven industrial process monitoring and diagnosis," *Annu. Rev. Control*, vol. 36, no. 2, pp. 220–234, 2012.

[102] Z. Ge, Z. Song, and F. Gao, "Review of recent research on data-based process monitoring," *Ind. Eng. Chem. Res.*, vol. 52, no. 10, pp. 3543–3562, 2013.

[103] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6418–6428, 2014.

[104] Z. Zhang and J. Zhao, "A deep belief network based fault diagnosis model for complex chemical processes," *Comput. Chem. Eng.*, vol. 107, pp. 395–407, 2017.

[105] H. Wu and J. Zhao, "Deep convolutional neural network model based chemical process fault diagnosis," *Comput. Chem. Eng.*, vol. 115, pp. 185–197, 2018.

[106] F. Cheng, Q. P. He, and J. Zhao, "A novel process monitoring approach based on variational recurrent autoencoder," *Comput. Chem. Eng.*, vol. 129, p. 106515, 2019.

[107] W. Sun, A. R. C. Paiva, P. Xu, A. Sundaram, and R. D. Braatz, "Fault detection and identification using bayesian recurrent neural networks," *Comput. Chem. Eng.*, vol. 141, p. 106991, 2020.

[108] S. Sun and J. Ren, "GASF-MSNN: A new fault diagnosis model for spatiotemporal information extraction," *Ind. Eng. Chem. Res.*, vol. 60, no. 17, pp. 6235–6248, 2021.

[109] M. Bauer, J. W. Cox, M. H. Caveness, J. J. Downs, and N. F. Thornhill, "Finding the direction of disturbance propagation in a chemical process using transfer entropy," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 1, pp. 12–21, 2006.

[110] J. Yu and M. M. Rashid, "A novel dynamic bayesian network-based networked process monitoring approach for fault detection, propagation identification, and root cause diagnosis," *AIChE J.*, vol. 59, no. 7, pp. 2348–2365, 2013.

[111] X. Chen, J. Wang, and J. Zhou, "Probability density estimation and bayesian causal analysis based fault detection and root identification," *Ind. Eng. Chem. Res.*, vol. 57, no. 43, pp. 14 656–14 664, 2018.

[112] J. Mori, V. Mahalec, and J. Yu, "Identification of probabilistic graphical network model for root-cause diagnosis in industrial processes," *Comput. Chem. Eng.*, vol. 71, pp. 171–209, 2014.

[113] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[114] T. A. Johansen, "Approximate explicit receding horizon control of constrained nonlinear systems," *Automatica*, vol. 40, no. 2, pp. 293–300, 2004.

[115] T. Parisini and R. Zoppoli, "A receding-horizon regulator for nonlinear systems and a neural approximation," *Automatica*, vol. 31, no. 10, pp. 1443–1451, 1995.

[116] S. W. Chen, T. Wang, N. Atanasov, V. Kumar, and M. Morari, "Large scale model predictive control with neural networks and primal active sets," *Automatica*, vol. 135, p. 109947, 2022.

[117] P. Kumar, J. B. Rawlings, and S. J. Wright, "Industrial, large-scale model predictive control with structured neural networks," *Comput. Chem. Eng.*, vol. 150, p. 107291, 2021.

[118] M. C. Campi, A. Lecchini, and S. M. Savaresi, "Virtual reference feedback tuning: a direct method for the design of feedback controllers," *Automatica*, vol. 38, no. 8, pp. 1337–1346, 2002.

[119] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, "Iterative feedback tuning: theory and applications," *IEEE Control Syst. Mag.*, vol. 18, no. 4, pp. 26–41, 1998.

[120] H. Hjalmarsson, "Iterative feedback tuning—an overview," *Intl. J. Adapt. Control Signal Process.*, vol. 16, no. 5, pp. 373–395, 2002.

[121] W. Tang and P. Daoutidis, "Input–output data-driven control through dissipativity learning," in *Am. Control Conf. (ACC)*. IEEE, 2019, pp. 4217–4222.

[122] ——, "Dissipativity learning control (DLC): A framework of input–output data-driven control," *Comput. Chem. Eng.*, vol. 130, p. 106576, 2019.

[123] ——, "Dissipativity learning control (DLC): theoretical foundations of input–output data-driven model-free control," *Syst. Control Lett.*, vol. 147, p. 104831, 2021.

[124] D. Hill and P. Moylan, "The stability of nonlinear dissipative systems," *IEEE Trans. Autom. Control*, vol. 21, no. 5, pp. 708–711, 1976.

[125] A. A. Alonso and B. E. Ydstie, "Process systems, passivity and the second law of thermodynamics," *Comput. Chem. Eng.*, vol. 20, pp. S1119–S1124, 1996.

[126] K. M. Hangos, J. Bokor, and G. Szederkényi, "Hamiltonian view on process systems," *AIChE J.*, vol. 47, no. 8, pp. 1819–1831, 2001.

[127] D. Hioe, J. Bao, and B. E. Ydstie, "Dissipativity analysis for networks of process systems," *Comput. Chem. Eng.*, vol. 50, pp. 207–219, 2013.

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

[129] J. Coulson, J. Lygeros, and F. Dörfler, "Data-enabled predictive control: In the shallows of the DeePC," in *Proc. 18th Eur. Control Conf. (ECC)*. IEEE, 2019, pp. 307–312.

[130] W. Favoreel, B. De Moor, P. Van Overschee, and M. Gevers, "Model-free subspace-based LQG-design," in *Am. Control Conf. (ACC)*. IEEE, 1999, pp. 3372–3376.

[131] R. Kadali, B. Huang, and A. Rossiter, "A data driven subspace approach to predictive controller design," *Control Eng. Pract.*, vol. 11, no. 3, pp. 261–278, 2003.

[132] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal control*. John Wiley & Sons, 2012.

[133] W. B. Powell, *Approximate dynamic programming: Solving the curses of dimensionality*. John Wiley & Sons, 2007.

[134] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.

[135] J. Kim and I. Yang, "Hamilton-Jacobi-Bellman equations for Q-learning in continuous time," *J. Mach. Learn. Res.*, vol. 120, pp. 739–748, 2020.

[136] B. Luo, H.-N. Wu, T. Huang, and D. Liu, "Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design," *Automatica*, vol. 50, no. 12, pp. 3281–3290, 2014.

[137] W. Tang and P. Daoutidis, "Distributed adaptive dynamic programming for data-driven optimal control," *Syst. Control Lett.*, vol. 120, pp. 36–43, 2018.

[138] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2nd ed. MIT Press, 2018.

[139] G. A. Rummery and M. Niranjan, "On-line Q-learning using connectionist systems," Cambridge University, Engineering Department, Tech. Rep., 1994.

[140] J. Shin, T. A. Badgwell, K.-H. Liu, and J. H. Lee, "Reinforcement learning – overview of recent progress and implications for process control," *Comput. Chem. Eng.*, vol. 127, pp. 282–294, 2019.

[141] S. Spielberg, A. Tulsyan, N. P. Lawrence, P. D. Loewen, and R. B. Gopaluni, "Toward self-driving processes: A deep reinforcement learning approach to control," *AIChE J.*, vol. 65, no. 10, p. e16689, 2019.

[142] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A Lyapunov-based approach to safe reinforcement learning," in *Proc. 31st Conf. Adv. Neural Inform. Process. Syst. (NeurIPS 2018)*, 2018.

[143] T. Xu, Y. Liang, and G. Lan, "CRPO: A new approach for safe reinforcement learning with convergence guarantee," in *Proc. 38th Int. Conf. Mach. Learn. (ICML)*. PMLR, 2021, pp. 11 480–11 491.

[144] F. Lin, M. Fardad, and M. R. Jovanović, "Design of optimal sparse feedback gains via the alternating direction method of multipliers," *IEEE Trans. Autom. Control*, vol. 58, no. 9, pp. 2426–2431, 2013.

[145] Y.-S. Wang, N. Matni, and J. C. Doyle, "Separable and localized system-level synthesis for large-scale systems," *IEEE Trans. Autom. Control*, vol. 63, no. 12, pp. 4234–4249, 2018.

[146] A. Themelis and P. Patrinos, "Douglas-Rachford splitting and ADMM for nonconvex optimization: Tight convergence results," *SIAM J. Optim.*, vol. 30, no. 1, pp. 149–181, 2020.

[147] W. Tang and P. Daoutidis, "Fast and stable nonconvex constrained distributed optimization: the ELLADA algorithm," *Optim. Eng.*, 2021, in press, doi:10.1007/s11081-020-09585-w.

[148] ——, "Coordinating distributed MPC efficiently on a plantwide scale: The Lyapunov envelope algorithm," *Comput. Chem. Eng.*, vol. 155, p. 107532, 2021.
[149] M. Arcak, C. Meissen, and A. Packard, *Networks of dissipative systems: Compositional certification of stability, performance, and safety*. Springer, 2016.
[150] P. H. Constantino, W. Tang, and P. Daoutidis, "Topology effects on sparse control of complex networks with Laplacian dynamics," *Sci. Rep.*, vol. 9, no. 1, pp. 1–9, 2019.
[151] P. Daoutidis, W. Tang, and S. S. Jogwar, "Decomposing complex plants for distributed control: Perspectives from network theory," *Comput. Chem. Eng.*, vol. 114, pp. 43–51, 2018.
[152] P. Daoutidis, W. Tang, and A. Allman, "Decomposition of control and optimization problems by network structure: Concepts, methods, and inspirations from biology," *AIChE J.*, vol. 65, no. 10, p. e16708, 2019.
[153] D. B. Pourkargar, A. Almansoori, and P. Daoutidis, "Impact of decomposition on distributed model predictive control: A process network case study," *Ind. Eng. Chem. Res.*, vol. 56, no. 34, pp. 9606–9616, 2017.
[154] A. Narasingam and J. S.-I. Kwon, "Koopman Lyapunov-based model predictive control of nonlinear chemical process systems," *AIChE J.*, vol. 65, no. 11, p. e16743, 2019.
[155] R. Huang, V. M. Zavala, and L. T. Biegler, "Advanced step nonlinear model predictive control for air separation units," *J. Process Control*, vol. 19, no. 4, pp. 678–685, 2009.
[156] E. Aydin, D. Bonvin, and K. Sundmacher, "NMPC using Pontryagin's minimum principle – application to a two-phase semi-batch hydroformylation reactor under uncertainty," *Comput. Chem. Eng.*, vol. 108, pp. 47–56, 2018.
[157] A. Koch, J. Berberich, J. Köhler, and F. Allgöwer, "Determining optimal input–output properties: A data-driven approach," *Automatica*, vol. 134, p. 109906, 2021.
[158] A. Koch, J. M. Montenbruck, and F. Allgöwer, "Sampling strategies for data-driven inference of input-output system properties," *IEEE Trans. Autom. Control*, vol. 66, no. 3, pp. 1144–1159, 2021.
[159] C. De Persis and P. Tesi, "Formulas for data-driven control: Stabilization, optimality and robustness," *IEEE Trans. Autom. Control*, vol. 65, no. 3, pp. 909–924, 2020.
[160] I. Markovsky, "Data-driven simulation of generalized bilinear systems via linear time-invariant embedding," *IEEE Trans. Autom. Control*, 2022, in press, doi:10.1109/TAC.2022.3146726.