A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies

Adam Thelen[1], Xiaoge Zhang[2], Olga Fink[3], Yan Lu[4], Sayan Ghosh[5], Byeng D. Youn[6], Michael D. Todd[7], Sankaran Mahadevan[8], Chao Hu[1*] and Zhen Hu[9*]

[1]Department of Mechanical Engineering, Iowa State University, Ames, 50011, IA, USA.
[2]Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong.
[3]Intelligent Maintenance and Operations Systems, EPFL, Lausanne, 12309, Switzerland.
[4]Information Modeling and Testing Group, National Institute of Standards and Technology, Gaithersburg, 20877, MD, USA.
[5]Probabilistic Design and Optimization group, GE Research, Niskayuna, 12309, NY, USA.
[6]Department of Mechanical Engineering, Seoul National University, Gwanak-gu, 151-742, Seoul, Republic of Korea.
[7]Department of Structural Engineering, University of California, San Diego, La Jolla, 92093, CA, USA.
[8]Department of Civil and Environmental Engineering, Vanderbilt University, Nashville, 37235, TN, USA.
[9]Department of Industrial and Manufacturing Systems Engineering, University of Michigan-Dearborn, Dearborn, 48128, MI, USA.

*Corresponding author(s). E-mail(s): chaohu@iastate.edu; zhennhu@umich.edu;
Contributing authors: acthelen@iastate.edu; xiaoge.zhang@polyu.edu.hk; olga.fink@epfl.ch; yan.lu@nist.gov; sayan.ghosh1@ge.com; bdyoun@snu.ac.kr; mdtodd@eng.ucsd.edu; sankaran.mahadevan@vanderbilt.edu;

Abstract

As an emerging technology in the era of Industry 4.0, digital twin is gaining unprecedented attention because of its promise to further optimize process design, quality control, health monitoring, decision and policy making, and more, by comprehensively modeling the physical world as a group of interconnected digital models. In a two-part series of papers, we examine the fundamental role of different modeling techniques, twinning enabling technologies, and uncertainty quantification and optimization methods commonly used in digital twins. This first paper presents a thorough literature review of digital twin trends across many disciplines currently pursuing this area of research. Then, digital twin modeling and twinning enabling technologies are further analyzed by classifying them into two main categories: physical-to-virtual, and virtual-to-physical, based on the direction in which data flows. Finally, this paper provides perspectives on the trajectory of digital twin technology over the next decade, and introduces a few emerging areas of research which will likely be of great use in future digital twin research. In part two of this review, the role of uncertainty quantification and optimization are discussed, a battery digital twin is demonstrated, and more perspectives on the future of digital twin are shared.

Keywords: Digital twin; Optimization; Machine learning; Enabling technology; Perspective; Industry 4.0, Review

1 Introduction

This paper is the first in a series of two that analyze the roles of modeling and twinning enabling technologies, uncertainty quantification (UQ), and optimization in digital twins. Modeling and twinning enabling technologies are fundamental methods used to bridge the information gap between a physical system and its digital counterpart. In this paper, we review and analyze many methods and modeling techniques currently used in digital twins, and classify them into groups based on the direction in which data flows.

We begin by reviewing an existing definition of digital twin. A digital twin, as originally proposed by Grieves (2014), is a virtual representation of a complex physical asset in the digital space for the purpose of closely characterizing the operations of the original physical process or system. The precise representation of the physical system in cyberspace is enabled by continuous data synchronization and information exchange between the digital twin and the physical counterpart. Having a digital replica of the physical entity of interest leads to enormous benefits spanning its entire life cycle, including the development phase (e.g., product design, resource planning, manufacturing process design), manufacturing phase (e.g., production process planning, manufacturing process control, maintenance of manufacturing equipment), service phase (e.g., performance and health monitoring, maintenance and control of fielded products, path planning), and disposal phase (e.g., end-of-life reuse, remanufacturing, and recycling). With the ability to conduct what-if analyses in the digital space, the benefits of digital twin technology are already materializing in a wide range of applications ranging from high-value manufacturing industry (Ayerbe et al., 2021) to personalized medicine (Corral-Acero et al., 2020), oil refinery management (Min et al., 2019), risk identification, and city planning (Lu et al., 2020a).

The digital twin concept centers around "individualized" digital models that capture the unique characteristics of individual products or process units. These models allow decision making to be optimized for each product or process unit, rather than based on the average characteristics of the entire population. This emerging technology poses new and challenging optimization problems at the forefront of model-based design, smart manufacturing, Industrial Internet of Things (IIoT), machine learning (ML), and predictive maintenance. The industry-scale adoption of the digital twin concept entails creating novel optimization solutions that use data collected from sensors and inspections (physical to digital) to provide decision makers with actionable information (digital to physical), thereby closing the digitalization loop. Major benefits include the ability to optimize control/maintenance actions to individual units and the potential to optimize the design of next-generation products.

The promise of digital twin concept and the need to guide the effective development of digital twin technology have motivated researchers to put together many review papers on digital twin. For example, motivated by the need to consolidate the many definitions and characteristics of digital twin, several review papers have been reported in recent years, such as Barricelli et al. (2019); Jones et al. (2020); VanDerHorn and Mahadevan (2021); Liu et al. (2021b). These papers summarized various definitions of digital twin and suggested generalized definitions and characteristics to distinguish digital twin from other models. Lim et al. (2020a) presented a comprehensive review of digital twin by considering its benefits to engineering product life cycle management and business innovation. Qi et al. (2021) surveyed enabling technologies and tools for digital twins and proposed a five-dimensional digital twin model. Fuller et al. (2020) analyzed enabling technologies for artificial intelligence, Internet of Things (IoT), and digital twins. Rasheed et al. (2020) reviewed the latest methodologies and techniques related to the construction of digital twins from the modeling perspective.

There are also numerous application-specific literature review papers. For instance, Jiang et al. (2021) reviewed the implementation of digital twins in civil engineering and discussed several problems that need to be addressed. Xie et al. (2021) discussed applications of digital twins at different states of a cutting tool's life cycle. Errandonea et al. (2020) reviewed the applications of digital twins for maintenance. Kritzinger et al. (2018) provided a categorical review of digital twins in manufacturing and grouped applications of digital twins into different categories according to the level of integration. Focusing on a similar topic, Cimino et al. (2019) reviewed the applications of digital twins in manufacturing. Along the same line, Ayerbe et al. (2021) discussed recent advances in the digitalization of lithium-ion battery cell manufacturing processes and proposed a three-layer structure of a digital twin of large-scale battery manufacturing. Ozturk (2021) conducted a literature survey over the applications of digital twins in the architectural, engineering, construction, operation, and facility management industries. Wagg et al. (2020) summarized the state-of-the-art and discussed future research directions of digital twin, focusing on applications in engineering dynamics. A survey of digital twins for verification and validation of industrial automation systems is available in Löcklin et al. (2020). Tao et al. (2018) presented a comprehensive review of digital twins in industry. He and Bai (2021) reviewed applications of digital twins for intelligent and sustainable manufacturing. Boje et al. (2020) discussed the future research directions of digital twin in construction engineering. Uhlemann et al. (2017) discussed the importance of digital twins in realizing cyber-physical production systems for Industry 4.0.

While current review papers cover a good amount of literature on different aspects of digital twin, such as definitions, enabling technologies, and applications in different domains, not all important aspects of digital twin have been covered in the current literature. Digital twin technology is an emerging technology that exploits the synergy between several key enabling technologies, including multiphysics simulations, sensing, ML, UQ, data analytics, to name a few. Modeling and twinning methods are key to integrating different technologies in a digital twin. However, a comprehensive review of the modeling and twinning methods and the associated UQ and optimization methods enabling digital twins is still missing. Such a review is essential because it may guide effective, industry-scale implementations of digital twins for different types of physical systems. Based on a literature review of over 230 research papers on digital twins, this paper aims to fill this void by (1) categorizing the modeling and twinning methods according to their roles in digital twins, and (2) providing a detailed discussion on the importance of UQ and optimization in the modeling and twinning methods of digital twins. The challenges and future research directions on digital twin will also be discussed based on the literature survey and lessons learned from case studies.

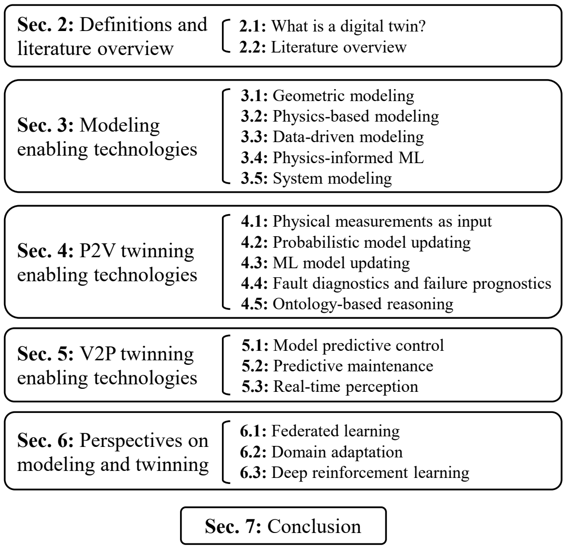

Sec. 2: Definitions and literature overview
2.1: What is a digital twin?
2.2: Literature overview
Sec. 3: Modeling enabling technologies
3.1: Geometric modeling
3.2: Physics-based modeling
3.3: Data-driven modeling
3.4: Physics-informed ML
3.5: System modeling
Sec. 4: P2V twinning enabling technologies
4.1: Physical measurements as input
4.2: Probabilistic model updating
4.3: ML model updating
4.4: Fault diagnostics and failure prognostics
4.5: Ontology-based reasoning
Sec. 5: V2P twinning enabling technologies
5.1: Model predictive control
5.2: Predictive maintenance
5.3: Real-time perception
Sec. 6: Perspectives on modeling and twinning
6.1: Federated learning
6.2: Domain adaptation
6.3: Deep reinforcement learning
Sec. 7: Conclusion

Fig. 1: Overview of topics covered in Part 1

In consideration of the large number of subjects covered, we split the review paper into two parts to make it easier to read. The first paper is an introduction to various definitions of digital twin in the literature and our proposed five-dimensional definition of digital twin. In addition, the first paper also covers commonly used modeling and twinning technologies to map a physical system to its digital counterpart (P2V) and the actions/decisions sent back by the digital twin that can be taken in the physical system (V2P). Figure 1 depicts the overall structure of Part 1 of the review paper.

Part 2 of the review paper (Thelen et al., 2022b) centers around describing the roles of UQ and optimization in digital twins so as to support a better characterization of physical systems in the digital space. This second part of the review also presents a case study on battery health management to showcase the specific implementation of a battery digital twin and the power of UQ and optimization in the digital twin in an end-to-end fashion. Additionally, Part 2 reviews the applications of digital twins at an industrial scale and available open-source tools and datasets related to digital twins, followed by a discussion on challenges and future research directions.

The remainder of this paper is organized as follows. Sec. 2 first discusses a variety of digital twin definitions reported in the literature and then conducts a comprehensive analysis on the trend of digital twin research. Sec. 3 categorizes the modeling enabling technologies in digital twins. Sec. 4 summarizes commonly used approaches for physical-to-virtual twinning. Sec. 5 elaborates the actions informed by virtual-to-physical connection that can be carried out at various stages of a physical system. Sec. 6 presents perspectives on modeling and twinning in digital twins. Sec. 7 concludes the first part of the two-part review paper.

2 Definitions of digital twin and literature overview

2.1 What is a digital twin?

According to the literature and other documents, the concept of "digital twin" was first implemented in the NASA Apollo 13 program in the 1960s. A digital twin model of Apollo 13 was created on earth to allow engineers on the ground to test possible solutions for a rescue mission in space (Barricelli et al., 2019; Nguyen et al., 2021). Michael Grieves made the first documented definition of digital twin in his presentation in the context of product life cycle management in 2003 and later in a white paper (Grieves, 2014). Since Grieves's presentation on digital twin in 2003, various digital twin concepts have been proposed. In a 2012 conference paper, Edward H. Glaessgen at NASA's Langley Research Center and David S. Stargel at the U.S. Air Force Office of Scientific Research gave the first specific definition of digital twin in the aerospace domain as:

"A Digital Twin is an integrated multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin (Glaessgen and Stargel, 2012)."

Since the first definition in the aerospace domain, digital twin has received increasing attention in the past decade in various application domains, such as manufacturing, automobile, energy, and civil engineering. Due to the broad applications of digital twin in different domains, numerous application-specific digital twin definitions have been proposed to capture the nature of digital twin in these domains. Aiming to reduce the risk of diluting the digital twin concept by a large number of definitions, efforts have been made in recent years to consolidate digital twin definitions and terminologies in several review papers (Barricelli et al., 2019; Jones et al., 2020; VanDerHorn and Mahadevan, 2021; Liu et al., 2021b). In particular, VanDerHorn and Mahadevan (2021) proposed a consolidated and generalized definition of digital twin as *"a virtual representation of a physical system (and its associated environment and processes) that is updated through the exchange of information between the physical and virtual systems"*. Amongst the many definitions of digital twin, the one specified in Glaessgen and Stargel (2012) is considered one of the most broadly accepted definitions, even though the definition may vary from application to application.

The distinction between a digital model, a digital shadow, and a digital twin is unclear in many cases. Further, having a large number of vague and inconsistent definitions of digital twin circulating in the literature may adversely affect industry interest in the adoption of this technology, creating a barrier to unleashing the maximum potential of the digital twin technology (Wright and Davidson, 2020). In what follows, we attempt to distinguish these three terms. It is argued that, for a digital model, data flow between the physical space and virtual space is optional, or at the very most, achieved manually as shown in Fig. 2. For a digital shadow, data flow is unidirectional from physical to digital. But for digital twin, the data flow has to be bidirectional (see Fig. 2) (Kritzinger et al., 2018). When digital twin is used in control-related applications, the bidirectional data flow needs to be automatic, often enabled by monitoring and control software. For the application of digital twin to support decision making, such as predictive maintenance, which will be reviewed in Sec. 5.2, however, the data flow from virtual to physical involves humans in the loop who carry out maintenance actions and, therefore, is not fully automatic but should be handled on time to avoid unexpected breakdown. Moreover, Zhang et al. (2021a) pointed out that since a physical entity usually has many different aspects that can be modeled, it is always necessary to clarify for what aspect the digital twin is constructed. Take a simple mechanical shaft as an example. The shaft may fail due to fatigue, extreme torque, etc. Since it is difficult for the digital space to mirror all aspects of the physical space, a digital twin is always constructed for specific aspect(s) relevant to the engineering problem the digital twin is used to solve, and these aspect(s) need to be specified before the construction of the digital twin.

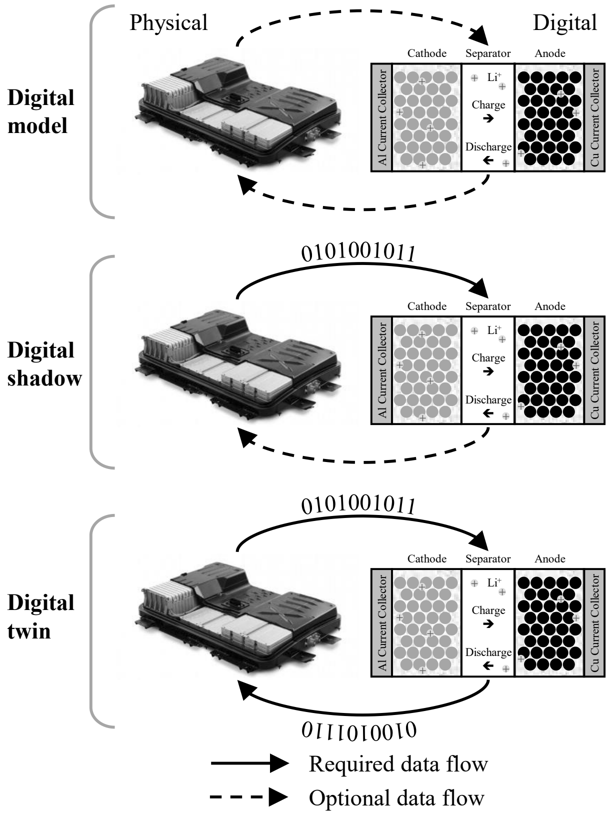


Fig. 2: Digital model, digital shadow, and digital twin concept for a lithium-ion battery cell within a larger pack

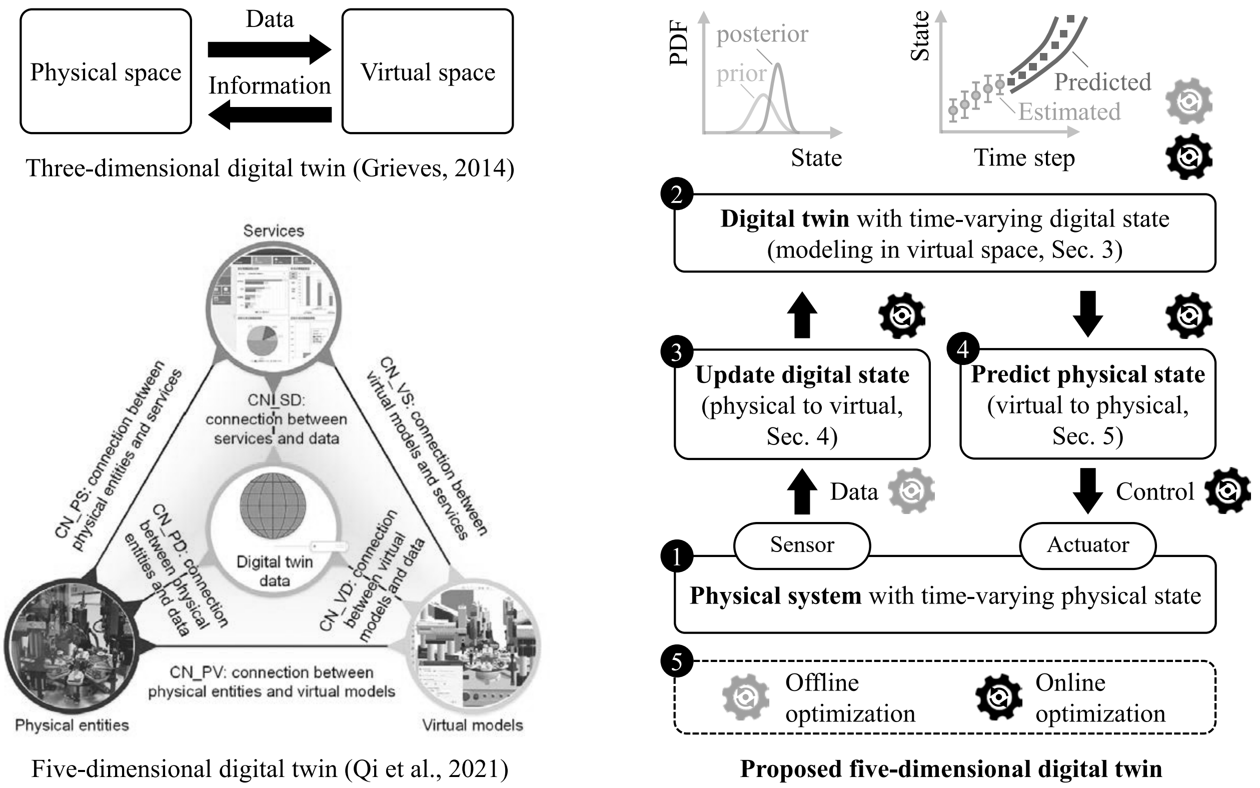
In addition, various digital twin models have been proposed in the past few decades. Two highly cited digital twin models in the literature are the three-dimensional digital twin model proposed by Grieves (2014) and the five-dimensional digital twin model suggested by Tao et al. (2018). As shown in Fig. 3, the three-dimensional digital twin model consists of a physical space, a virtual space, and connections between them. Tao et al. (2018) expands the three-dimensional digital twin model into a five-dimensional digital twin model for manufacturing, which includes a real machine, a virtual machine, a services component, digital twin data, and the connections as depicted in Fig. 3.

Based on the definition of digital twin given in (Glaessgen and Stargel, 2012) as mentioned above, we suggest a new five-dimensional digital twin model as illustrated in Fig. 3. The proposed five-dimensional digital twin model can be considered an extended version of the three-dimensional model and is defined as follows

$$\mathrm{DT} = \mathbb{F}(\mathrm{PS},\ \mathrm{DS},\ \mathrm{P2V},\ \mathrm{V2P},\ \mathrm{OPT}) \tag{1}$$

As indicated in the equation above, the proposed five-dimensional digital twin model consists of *a physical system (PS)*, *a digital system (DS)*, *an updating engine (P2V)*, *a prediction engine (V2P)*, and *an optimization dimension (OPT)*, and $\mathbb{F}(\cdot)$ integrates all five dimensions together to be an effective digital twin. The physical system (PS) with sensing capability gathers data from multiple sources using various sensing and data acquisition techniques. The updating engine (P2V) updates the state of the digital model based on the sensor data. The updated digital models (DS) are then employed to predict the future state of the physical system using the prediction engine (V2P), thereby enabling predictive decision making feeding back to the physical system through actuators (control, maintenance, path planning, etc.). Finally, an optimization dimension (OPT, i.e., the fifth dimension) supports the functionalities of the other four dimensions in the digital twin by optimizing the data collection, modeling, state estimation, decision making, etc. The optimization dimension consists of two elements, namely offline optimization and online optimization. These two optimization elements will be explained in detail in Part 2 of the review paper (Thelen et al., 2022b). The seamless integration of the five dimensions (PS, DS, P2V, V2P, and OPT) enables digital twins to fulfill the need for a real-time mirror of the life cycle of a physical system, thereby supporting optimal decision making.

To highlight how a five-dimensional digital twin might operate in practice, we can look at the example of a digital twin for a lithium-ion battery cell within a battery pack, shown in Fig. 2. When designing a lithium-ion battery pack, engineers are most concerned with managing the rate and amount of degradation of each cell in the


Physical space
Data
Information
Virtual space
Services
CN_SD: connection between services and data
CN_PS: connection between physical entities and services
CN_VS: connection between virtual models and services
CN_PD: connection between physical entities and data
CN_VD: connection between virtual models and data
Digital twin data
CN_PV: connection between physical entities and virtual models
Physical entities
Virtual models
PDF
posterior
prior
State
State
Predicted
Estimated
Time step
2
Digital twin with time-varying digital state (modeling in virtual space, Sec. 3)
3
Update digital state (physical to virtual, Sec. 4)
4
Predict physical state (virtual to physical, Sec. 5)
Data
Control
Sensor
Actuator
1
Physical system with time-varying physical state
5
Offline optimization
Online optimization

Three-dimensional digital twin (Grieves, 2014)

Five-dimensional digital twin (Qi et al., 2021)

Proposed five-dimensional digital twin

Fig. 3: Comparison of different digital twin (DT) models and the proposed model

pack driven by the cell's usage (e.g., charging and discharging rates, depth of discharge, and temperature). In this scenario, a digital twin model of each cell within a pack can be constructed to track and forecast the capacity degradation trajectory of the cell and ultimately drive decision making around when the pack needs to be retired from service. Sensor measurements of the cell's state of health (PS) can be used to update (P2V, OPT) the digital model (DS) of the cell. Then, the digital model (DS) can be used to predict the cell's future state of health and degradation trajectory (V2P). Engineers can then use the predicted future degradation trajectory to inform decision making about the optimal time when the pack may need to be retired from service (V2P, OPT). The complete flow of data through all five digital twin dimensions outlined in Fig. 3 is what ultimately characterizes this type of workflow as a true digital twin. This digital twin application for forecasting the future health of a lithium-ion cell and using the forecast health for predictive maintenance is extensively investigated as a case study in Part 2 of this review.

With a focus on the modeling and twinning technologies (i.e., the updating and prediction engines in Fig. 3), in the subsequent sections, we first perform an extensive literature review of the state-of-the-art and then classify the enabling technologies into distinct categories. The role of the optimization dimension in digital twin and future research directions will also be discussed.

2.2 Literature overview

Our literature review starts with a Google scholar search using the keyword "digital twin". In addition to the review papers summarized in Sec. 1, 230 research papers, excluding papers from predatory journals, were reviewed. The majority of these 230 research papers were published between 2017 and 2022, with some published between 2011 and 2012. They are classified into different groups by application domain as shown in Fig. 4. This figure indicates that manufacturing, followed by mechanical, civil, and structures, accounts for most of the digital twin papers. 107 out of 230 papers come from the manufacturing domain,

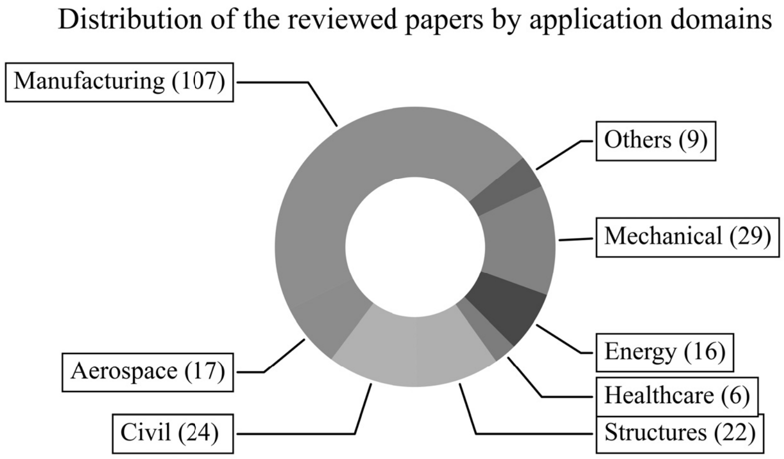


Fig. 4: Distribution of reviewed papers by domain ("Civil" includes civil engineering, building engineering, smart city, and transportation; "Mechanical" includes mechanical engineering, automotive, robotics, and marine; "Others" includes supply chain, optical communication system, and agricultural engineering.

which implies a leading role of manufacturing in the digital twin field.

The following information is then extracted from the literature survey:

- **Top-cited papers**, which are papers with an average number of citations per year greater than 25 (the average number of citations per year to a paper is its total number of citations divided by the number of years elapsed since the publication of the paper).
- **Major journals**, which are the journals where research papers on digital twin are frequently published based on the 230 research papers reviewed.
- **Publicly available tools/datasets**: tools and datasets related to digital twin from the 230 papers and associated websites.
- **Modeling methods**: methods used in the papers to create the virtual space.
- **Twinning methods**: methods used to establish physical-to-virtual and virtual-to-physical connections.
- **Optimization**: roles of optimization in the digital twin models presented in the papers.

Next, we will summarize the major findings related to the first two items listed above. The methods of modeling, twinning (i.e., physical-to-virtual and virtual-to-physical), and optimization will be summarized and classified into different categories in Sec. 3. The publicly available tools/datasets will be provided in Sec. 4.2 of Part 2 of the review paper.

2.2.1 Top-cited papers and trends in different domains

From the 230 research papers, we first identify papers whose average number of citations per year is greater than 25. It is found that many papers in the manufacturing domain satisfy this criterion. It is not entirely surprising since digital twin has been a hot topic in smart manufacturing. We, therefore, list the top-cited papers in the manufacturing domain separately from those in the other domains.

Table 1 summarizes the top ten most-cited papers in the manufacturing domain from the 230 reviewed papers, based on the average number of citations per year. Qi and Tao are the the most highly-cited researchers in this field at this time (Tao et al., 2018; Qi and Tao, 2018; Tao and Zhang, 2017; Qi et al., 2018). Most of the top-cited papers in the manufacturing domain focus on digital twins of manufacturing shop-floors, such as assembly, production management and control, and configuration optimization. There is one paper that focuses on geometry assurance in individualized production. Table 2 lists the top-cited papers from the civil, aerospace, energy, and healthcare domains, which reflect the trends of digital twin outside the manufacturing sector. As

Table 1: Top ten top-cited papers in manufacturing domain (Based on yearly average number of Google scholar citations as of April 2022)

Title	**Author and Publication year**	**Citations**	**Application**
Digital twin-driven product design, manufacturing and service with big data	Tao et al. (2018)	1435	Design and Manufacturing
Shaping the digital twin for design and production engineering	Schleich et al. (2017)	719	Production
Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison	Qi and Tao (2018)	789	Smart manufacturing
Digital twin shop-floor: a new shop-floor paradigm towards smart manufacturing	Tao and Zhang (2017)	585	Shop floor
Digital twin-based smart production management and control framework for the complex product assembly shop-floor	Zhuang et al. (2018)	313	Shop floor
Toward a Digital Twin for real-time geometry assurance in individualized production	Söderberg et al. (2017)	372	Geometry assurance
Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop	Leng et al. (2019)	235	Shop floor
Defining a digital twin-based cyber-physical production system for autonomous manufacturing in smart shop floors	Ding et al. (2019)	225	Shop floor
Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system	Liu et al. (2021c)	100	Configuration optimization
Digital twin service towards smart manufacturing	Qi et al. (2018)	231	Service in Manufacturing

shown in this table, aerospace, civil, and mechanical are leading the trend of digital twin outside the manufacturing domain. Five out of the 12 top-cited papers concentrate on fault diagnostics and prognostics of physical assets using digital twin (Xu et al., 2019; Li et al., 2017; Glaessgen and Stargel, 2012; Jain et al., 2019; Tuegel et al., 2011). The others focus on optimization, management, and service.

Following that, Fig. 5 shows the cumulative number of papers over the past five years in each different application domain. It is clear that the number of papers in the manufacturing domain is much higher than the other domains. This agrees with the earlier analysis of the most highly cited papers which found that a majority of them were in the manufacturing domain. Note that the trends abruptly flatten from 2021 to 2022 because this review paper was written in early 2022 and did not account for papers published later in the year. It is expected that the increasing trend continue into 2022.

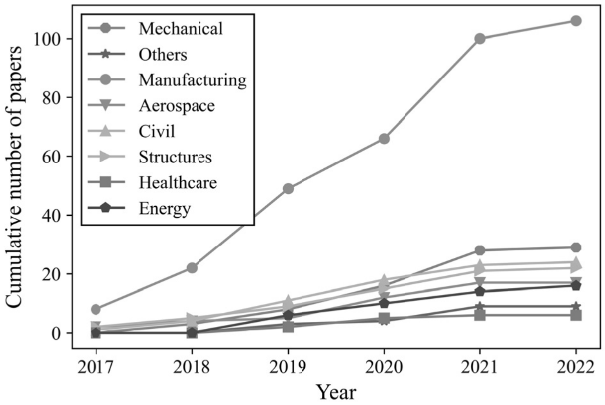


Fig. 5: Cumulative number of papers of different application domains

To better analyze the trends in the other domains, we remove the manufacturing data, and re-scale the plot. Fig. 6 shows the cumulative numbers of papers in the past five years, excluding manufacturing. In recent years, the cumulative number of papers in the other domains, especially civil, mechanical, and energy, have been growing rapidly.

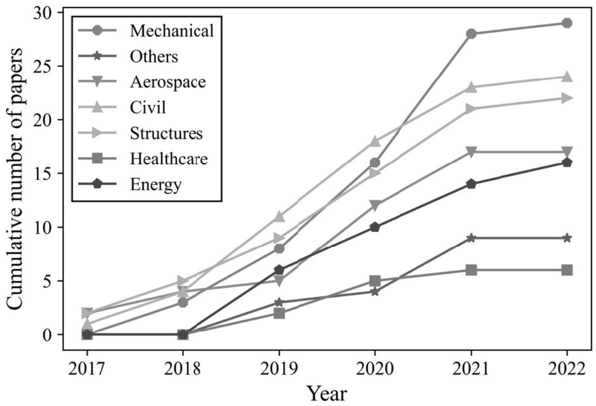


Fig. 6: Cumulative number of papers of different application domains excluding manufacturing

2.2.2 Major journals

Table 3 presents major journals where the 230 reviewed papers were published. Only journals with more than three published papers are listed. Shown in the table, manufacturing-related journals, namely *Journal of Manufacturing Systems*, *International Journal of Advanced Manufacturing Technology*, and *Robotics and Computer Integrated Manufacturing*, are leading journals. Other leading journals include *Journal of Ambient Intelligence and Humanized Computing*, *IEEE Access*, *Engineering Fracture Mechanics*, and *International Journal of Production Research*.

3 Modeling enabling technologies

Modeling plays an essential role in digital twins by creating a digital replica of the physical entity of interest. According to the purpose and complexity of different modeling methods, we classify them into the following five categories.

3.1 Geometric modeling

(a) Solid modeling

Computer-aided design (CAD) is a virtual solid modeling technique where 3D objects are created from a number of 2D drawings using computer software. Many individual components can be combined in an assembly to create a 3D solid model of a larger system. Solid models help designers and engineers see how a product will look and function in the real world. There are a number of free and paid-for 3D solid modeling software on the market. Each software package differs in its intended audience and scope of available features. For engineering, popular software includes Autodesk Inventor and AutoCAD, PTC Creo, SolidWorks, and Catia (Cai et al., 2020). Free software packages include Unity3D, Unreal Engine, and Blender (Leng et al., 2020; Matulis and Harvey, 2021; Liu et al., 2021e).

A simple but useful application of solid modeling for digital twin is to create models of all the equipment for a production line. Then, the modeled equipment can be moved around, checked for timing, or simulated consistent with a range of different goals. For example, Liu et al. (2021f) created a virtual workshop with various machining tools so that the order in which different parts were machined could be optimized. Similar work by Yi et al. (2021) investigated ideas for using 3D models of the equipment in an assembly line to optimize the layout for reduced travel distance between steps. Vathoopan et al. (2018) proposed creating a solid model of a robotic arm to detect and alarm when the robot's joints may have moved further than physically possible as predicted by the solid model. Ayani et al. (2018) created a complete solid model of an old analog-controlled production machine so that its movements, timings, and controls could be uploaded to the cloud and further studied. Huang et al. (2022a) modeled a CNC tool changing system and all the tools inside of it for the purpose of improving tool change speeds. Creating solid models for optimization problems which focus on the physical size and location of manufacturing equipment and tools is relatively easy because minor details like material properties and movable joints can be ignored. If, however, the solid model will be used for simulation of stresses, movements, or interference using physics-based simulations (see Sec. 3.2), the

Table 2: Top-cited papers in each application domain excluding manufacturing (based on yearly average number of Google Scholar citations as of April 2022)

Title	**Publication year**	**Citations**	**Application domain**
A digital-twin-assisted fault diagnosis using deep transfer learning	Xu et al. (2019)	154	Mechanical (Automotive)
The digital twin paradigm for future NASA and U.S. Air Force vehicles	Glaessgen and Stargel (2012)	1226	Aerospace
Reengineering aircraft structural life prediction using a digital twin	Tuegel et al. (2011)	639	Aerospace
Dynamic Bayesian network for aircraft wing health monitoring digital twin	Li et al. (2017)	171	Aerospace
Machine learning based digital twin framework for production optimization in petrochemical industry	Min et al. (2019)	141	Energy (Oil industry)
A digital twin approach for fault diagnosis in distributed photovoltaic systems	Jain et al. (2019)	106	Energy
A BIM-data mining integrated digital twin framework for advanced project management	Pan and Zhang (2021)	60	Civil (Construction)
Developing a digital twin at building and city levels: Case study of West Cambridge campus	Lu et al. (2020a)	82	Civil (Smart city)
Digital twin–Proof of concept	Haag and Anderl (2018)	288	Mechanical (Structural Health Monitoring)
An application framework of digital twin and its case study	Zheng et al. (2019)	206	Others (Service)
How to tell the difference between a model and a digital twin	Wright and Davidson (2020)	104	Others (Service)
A novel cloud-based framework for the elderly healthcare services using digital twin	Liu et al. (2019)	171	Healthcare

modeling process requires all details about the equipment be defined, increasing the modeling time substantially.

Another application of solid modeling in digital twin is simulating and studying human-environment interaction. Solid modeling enables researchers to investigate how humans fit into the digital twin environment. Vatankhah Barenji et al. (2021) investigated concepts of human-robot interaction and ways to improve energy savings. Pairet et al. (2019) proposed modeling an entire offshore oil drilling platform to study human-robot interaction in the dense 3D space. Similarly, Lohtander et al. (2018) created a 3D model of a human assembly area to study ergonomics and optimize the placement of part bins. This area of research is of great importance, especially in a manufacturing setting, where humans and robots coexist in a constrained environment. It is envisioned that solid modeling of environments will play a large role in designing smarter robots which can better interact safely in proximity to humans.

Other applications of solid modeling to digital twins include tolerancing, geometry variance control, and defect control in manufacturing settings. Digital twin models of a production process can be used to improve quality control in various ways. Liu et al. (2022) investigated integrating solid modeling and surface roughness prediction for adaptive manufacturing control. Schleich et al. (2017) proposed computer aided tolerancing and geometry variation control via a solid modeling approach. Zambal et al. (2018) discussed how

Table 3: Major journals on digital twin based on the reviewed papers

Journal Name	No of Papers
Journal of Manufacturing Systems	25
IEEE Access	10
International Journal of Advanced Manufacturing Technology	9
Robotics and Computer Integrated Manufacturing	6
International Journal of Computer Integrated Manufacturing	6
Procedia CIRP	6
CIRP Annuals - Manufacturing Technology	5
IFAC Papers Online	5
Journal of Intelligent Manufacturing	4
Journal of Ambient Intelligence and Humanized Computing	4
Engineering Fracture Mechanics	4
International Journal of Production Research	3
Advanced Engineering Informatics	3
Automation in Construction	3
Computer and Structures	3

one might make a solid model system of a carbon fiber production process so that deviations in the equipment or manufacturing process could be simulated. Through simulation, the effects of equipment deviations on the final finished carbon fiber product can be quantified and understood.

Solid modeling has many promising uses in digital twin applications. With the ability to model the geometry of any object, solid modeling enables additional optimization dimensions not captured through sensor measurements and other techniques. One of the key problems using solid models in digital twins is the excessive amount of time required to create models which can be used in accurate physics-based simulations, such as finite element analysis (FEA) or motion simulations in Sec. 3.2. In many cases, the solid models of the parts and machines researchers aim to use in digital twin simulations have already been created by the original equipment manufacturers (OEMs). Similar to ideas discussed in Grieves (2014) white paper, it is entirely conceivable that future high value assets (e.g., aircraft, spacecraft, industrial equipment, and even entire production factories) could be delivered to the customer accompanied by a complete digital solid model for use in a digital twin framework. In turn, the extensive time commitment the OEM has sunk into creating the initial 3D solid model of the product could now be recouped by supporting the end user with a long-term reliability and predictive maintenance contract. With lasting communication between the OEM and the end user through the online solid model, safety issues can be managed quicker, replacement parts can be identified more easily, and future upgrades can be tested and checked virtually for compatibility with a given customer's asset before committing to manufacturing the parts. Ultimately, the sharing of solid models between OEMs and end users is necessary to accelerate the adoption of digital twin technology.

(b) Laser scanning

Laser scanning, also commonly referred to as LiDAR (light detection and ranging), is a geometry modeling technique which uses laser light to repeatedly sample the distance to different points on an object's surface. The recorded measurements are virtually assembled into a 3D point cloud of (X, Y, Z) coordinate data to describe the surface morphology of the scanned object in great detail. Unlike solid modeling techniques which can model internal components, material changes, and geometries, laser scanning can only model surface-level features. However, laser scanning makes up for its inability to model internal geometries by generally being much faster at modeling objects. When it is paired with the appropriate software, laser scanning can create a point cloud of data points on a computer in near real-time.

Laser scanning has been used in many applications for digital twin. In the manufacturing domain, laser scanning has been proposed for quickly checking the geometry of traditionally manufactured or 3D printed parts (Almalki et al., 2022; Wärmefjord et al., 2017; White et al., 2021). Also in manufacturing, Dröder et al. (2018) showed a proof of concept for a digital twin considering human-robot interaction. The researchers used a Microsoft Kinect 2.0 LiDAR sensor to detect humans in a robotic arm cell. Based on the proximity of the human to the robotic arm, the robotic arm's path and speed were automatically optimized in real-time to create a safe environment for the human while not ceasing operation entirely. In contrast to other systems which use laser trip-wires that completely stop the robot when a human enters the area, the LiDAR-based digital twin system significantly improved manufacturing performance while maintaining a safe working environment. In a different application of laser scanning and robots, Wang et al. (2020a) laterally mounted an industrial camera on a robotic welder to take images of the process for improved real-time quality control via augmented reality data streaming. LiDAR and other laser scanning techniques will be extremely useful for the development of more complex manufacturing sensor networks connected to a digital twin. Real-time data streaming to and from the physical and virtual dimensions will enable more productive manufacturing lines without compromising worker safety.

The ability of laser scanning to quickly map and model large structures, landscapes, and buildings makes it particularly useful for digital twin modeling in the civil engineering domain (Lu et al., 2020b; Pan and Zhang, 2021). One of the most common uses of LiDAR and laser scanning for digital twin is for scanning the surface of large buildings (Deng et al., 2021), bridges (Shim et al., 2019), or other structures (Lim et al., 2020b). Specifically in Shim et al. (2019), the authors discussed how LiDAR laser scanning could be used to periodically model the surface features of different bridge components for preventative maintenance. In this system, the raw scanned point cloud data is fed in batches to the cloud where structure recognition deep learning (DL) algorithms sort the scanned images according to the type of structure it is. Then, within the digital twin network, the images are mapped into their respective ontological structure file (see Section 3.5 (b)) where they are further analyzed for cracks, pitting, shifting, or other undesirable degradation indicators. Any positively identified defects would then be loaded into a global solid model simulation for physics-based stress analysis. In the next section, we discuss such physics-based simulations, and their role in digital twins. Altogether, applications of LiDAR and laser scanning for digital twin modeling will become extremely important in the coming years as cities struggle to maintain the ever increasing number of bridges and other critical infrastructure.

(c) VR, AR, and MR technologies

In addition to solid modeling and laser scanning, other techniques, such as virtual reality (VR), augmented reality (AR), mixed reality (MR), have also been explored as viable methods of constructing digital replicas of various physical entities. These visualization and interaction techniques differ in how they display relevant information. The most simple method, AR, overlays relevant contextual data to the user by projecting over assets in the physical world. Typically, the projections are done using light projection onto surfaces, or in some cases, through a wearable headset with transparent lenses. MR technologies further extend the capabilities of AR by allowing interactions between the digital objects and the physical world. This technology is closely followed by VR, which is a completely immersive digital environment where interactions happen exclusively in the virtual world. These data visualization and interaction technologies are promising enablers of digital twin modeling because of their ability to connect users, assets, and data streaming in real-time. VR, AR, and MR technology research is an active field. For example, Tadeja et al. (2020) developed an immersive aerospace design environment built upon VR to aid the design process of aerodynamic components by interactively visualizing performance and geometry. Other work by Cai et al. (2020) leveraged AR to link the layout information between a physical and digital asset in a reconfigurable additive manufacturing system made of robotic arms for tool path planning and simulation. Choi et al. (2022) integrated MR in a digital twin to achieve safety-aware human-robot collaboration, where MR was combined with safety-related monitoring to track the shared workplace and real-time safety distance calculation. Research into VR, AR, and MR technology remains a hot and active topic, as the role of these powerful data visualization and interaction techniques is not yet fully understood. In the future, it is expected that these technologies will enable high-fidelity interactive digital twin models that can be used to visualize, control, and optimize various processes and designs in real-time.

3.2 Physics-based modeling

Scientists observe the basic principles of how our world works and translate them into physical laws.

These laws define the rules for the motion of matter and other physical phenomena through space and time. It is well-known that partial differential equations (PDEs) can mathematically describe certain physical laws. Two well-known examples of PDEs are the Navier–Stokes equations describing fluid motion Chorin et al. (1990) and the heat equation describing heat diffusion Widder (1976). Combining PDEs into a mathematical model and solving this model allows engineers and scientists to simulate and observe the physical phenomena governing a system or process. In computer-aided engineering (CAE), engineers and researchers use software packages to formulate mathematical models consisting of PDEs that describe the underlying physics, initial conditions, and boundary conditions. After defining the models, the software solves the PDEs to simulate the physical effects so that engineers and researchers can better understand the physical interactions. Simulating the solution approximates the input-output relationship of the physical system, where the input consists of the physical system properties, parameterized in the mathematical model, as well as the operating and environmental conditions, and the output is the performance of interest. Physics-based modeling software is diverse and complex in the range of applications it can be used for. Below we outline some of the most common types of physics engineers and researchers aim to simulate.

- **Types of physics**: Physics-based modeling and simulation for use in digital twins covers a wide scope. Below, we highlight and summarize some of the most common types of physics and physical modeling techniques used for digital twin.

  1. Solid body structural analysis of stress and strain is typically conducted using finite element analysis (FEA) software where the objects are modeled as a tessellation mesh of triangles, and the forces are calculated at each node in the mesh. FEA is flexible in that the resolution of the mesh can be tuned to balance the trade-off between simulation accuracy and computational costs. FEA has so far been an important modeling technique for digital twin (Li et al., 2017; Angjeliu et al., 2020; Millwater et al., 2019; Moi et al., 2020; Karve et al., 2020; Ye et al., 2020; Liu et al., 2021c; Bellalouna, 2021; Wang et al., 2022a). Specific examples of FEA for digital twin include stress analysis of a ground vehicle suspension (Hu and Youn, 2011), wear analysis of cutting tool in manufacturing (Zhang et al., 2021b), strain analysis (Revetria et al., 2019), and dynamic structural analysis of an aircraft wing (Seshadri and Krishnamurthy, 2017). As an initial digital twin proof of concept, Haag and Anderl (2018) demonstrated simultaneous simulation and measurement of forces in a beam bending test bench.
  2. Thermal and fluid flow analyses are typically conducted using computational fluid dynamics (CFD) software (Zhou et al., 2021). Fluid physics are especially relevant in aircraft and spacecraft simulation, where digital twin was first conceived (Grieves, 2014). In many cases, fluids are closely tied to heat transfer, as flowing fluids are exceptionally effective at exchanging heat. Fan et al. (2013) performed thermal analysis for a battery cooling system of an electric vehicle. In the future, thermal and fluid flow analyses will prove exceptionally useful in digital twins.
  3. Kinematic and dynamic analyses of mechanisms using multi-body dynamics (MBD) models are another type of physics commonly modeled. Ha et al. (2018) analyzed the motion of robots using MBD; Xia et al. (2021a) used MBD to simulate and test the interference and collisions of the components in their digital twin model.
  4. Multiphysics simulations simultaneously simulate multiple coupled physical phenomena by sharing information between individual physics-based simulations (DeLaurentis and Mavris, 2000; Hu and Mahadevan, 2018a). Examples include coupled FEA and CFD simulations in fluid-structure interaction for aircraft wings (Tezduyar and Osawa, 2001) and wind turbines (Hsu and Bazilevs, 2012), and coupled electrochemical-mechanical simulations for modeling Li-ion battery cells (Allu et al., 2014; Wang et al., 2022b). Another example of coupled multiphysics interaction is the hypersonic vehicle shown in Fig. 7. This multiphysics simulation features coupled aerodynamic and aeroelastic analyses to

predict the complex vehicle response under hypersonic airflow conditions (Culler and McNamara, 2010).

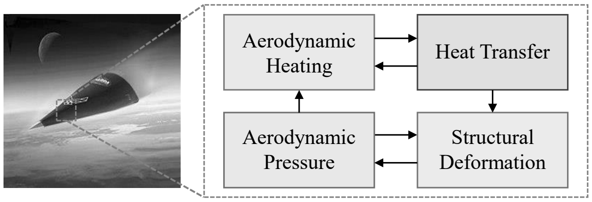


Fig. 7: Coupled multiphysics simulation of a hypersonic vehicle panel (Hu and Mahadevan, 2018a)

- **Levels of fidelity**: When building a physics-based model, it is important to consider the minimum level of detail (fidelity) required to approximate the physical input-output relationship in order to optimize the balance between computational costs and model accuracy. Take for example a fluid dynamics problem which can be solved using different CFD models. With the highest fidelity, direct numerical simulation can provide an accurate high-fidelity solution to model turbulence flow. With slightly lower fidelity, large eddy simulation and the Reynolds-averaged Navier–Stokes equations can also be used to solve the same problems more quickly, albeit with a slightly lower accuracy. Another example of multi-fidelity modeling arises in the battery modeling community. There is a well-established hierarchy of electrochemical-thermal models that consists of (from high- to low-fidelity) the 3D Doyle–Fuller–Newman (DFN) model, 2+1D DFN model, the DFN model (or pseudo-two-dimensional), and the single particle model Marquis et al. (2020). The field of multi-fidelity modeling is rich with literature, and interested readers should read several of the dedicated review papers (Fernández-Godino et al., 2016; Peherstorfer et al., 2018; Giselle Fernández-Godino et al., 2019). One final note regarding model fidelity is that the data-driven models presented in Sec. 3.3 fall into the broad category of low-fidelity models, and great effort is being done to find ways to combine physics-based and data-driven models for improved fidelity (see Sec. 3.4)

When defining a relevant physics-based modeling problem, the engineering problem at hand determines which types of physics should be modeled. Many commercial CAE software tools streamline the execution of three basic steps: (1) loading in 3D geometry data from a geometric model created in CAD software or via laser scanning as reviewed in Sec. 3.1, (2) defining the initial conditions and determining the geometric and physical bounds of the simulation, and (3) solving the model to approximate the performance of interest. Examples of software tools used to solve physics-based simulations are ANSYS Mechanical, Abaqus FEA, and Altair HyperMesh for FEA, ANSYS Fluent, Autodesk CFD, and SimScale for CFD, MATLAB & Simulink (Simscape Multibody), MSC ADAMS, and RecurDyn for MBD, and COMSOL Multiphysics and MSC Nastran for coupled multiphysics simulation. A few noteworthy open-source packages relevant to digital twin are OpenFOAM and Stanford University's Unstructured (SU2) package for CFD, MBDyn for MBD, and for multiphysics simulation, Advanced Simulation Library (ASL) and Chrono open-source options.

It is important to strike a balance between accuracy and computational cost when choosing the level of fidelity for a physics-based simulation. No matter how high the fidelity of a physics-based model might be, it is still only an approximation of reality because of missing physics, flawed assumptions, finite mesh discretization, etc. If the resulting model bias is large (likely for a low-fidelity model), it is important to quantify the bias offline and compensate for it online (see Sec. 2.1.2 in Part 2 of the review paper for a detailed discussion on the offline calibration of dynamic system models). Furthermore, it may be counter-intuitive to say that high-fidelity models are not always preferred even when computational costs incurred in DS, P2V, and V2P are not a concern. Practitioners need to keep in mind that high-fidelity models contain larger numbers of parameters that need to be calibrated offline, and during online deployment, a subset of the parameters need to be further tuned. Offline validation and online fine tuning requires the use of previously collected data and measurements. However, most of the time, very limited data is available for both offline model parameter estimation and online fine tuning. Practioners who deploy a high-fidelity model but have inadequate data to properly tune the parameters may find that model underperforms, and exhibits poor accuracy. As a result, practical implementations of physics-based models are (1) a lower-fidelity model that offers the desired accuracy at a lower cost and with a smaller number of parameters for offline calibration and (2) a small and critical subset of the model parameters are identified for online updating, reducing the amount of data required (see also a discussion in Sec. 4.2 (a)).

3.3 Data-driven modeling

Data-driven models are needed for modeling physics in a digital twin under either of two main scenarios:

1. the underlying physics is too complicated or is not fully understood, and as a result, building an accurate physics-based model is impossible; or
2. the physics is well understood and can be modeled using available software, but the simulation is too computationally expensive or time consuming to be useful in a digital twin, especially when many model runs are required (such as with UQ tasks).

In either scenario, data-driven models must be constructed to represent the underlying physical input/output behavior, enabling efficient prediction of digital states for real-time control and optimization in a digital twin. Shown in Fig. 8, data-driven models can be grouped into three classes according to their application domain, namely (1) data-driven models for *degradation modeling*, (2) data-driven *surrogate models* which can entirely replace the physics-based models discussed in Sec. 3.2, and (3) data-driven models for *dynamic system identification* using sensor measurements from a physical system (PS in Eq. (1)). Moreover, it is worth mentioning that system identification is fundamentally very similar to surrogate modeling. System identification is generally the determination of a model's characteristic parameters (regardless of model class) that minimize some predictive error metric, and one may note that essentially a surrogate model is trying to mimic a system's behavior also with a goal towards some minimized prediction error metric. The main difference between the two is that surrogate modeling is typically constructed and optimized using computer simulation data for any type of physical simulation at any time scale, while system identification mainly uses online sensor monitoring data and/or offline experimental data collected to tune the specific model's parameter(s). The close similarity is why many of the surrogate modeling methods and system identification methods can be used interchangeably. This relationship is further explained in Fig. 8.

Data-driven models for use in digital twins can be classified into two categories according to the fundamental theory and construction (e.g. numerical or analytical), namely *statistical models* and *ML models*. Statistical models and ML models have been used for both prediction and inference (Danilo et al., 2018). "*Statistical methods have a long-standing focus on inference, which is achieved through the creation and fitting of a project-specific probability model, while ML concentrates on prediction by using general-purpose learning algorithms to find patterns in often rich and unwieldy data*" (Danilo et al., 2018; Bzdok et al., 2017). In what follows, we discuss the latest data-driven modeling trends and provide insights on how each method can or has been used in a digital twin framework.

3.3.1 Statistical models

Statistical models play a vital role in establishing both P2V and V2P connections in digital twins. For P2V connection, statistical models have been used in the context of dynamic system identification to build state-space models based on measurement data. For V2P connection, statistical models have been extensively used to build degradation models for failure prognostics and predictive maintenance of physical systems.

- *Statistical models for dynamic system identification (P2V).* System identification is the process of learning a mathematical representation of a dynamic system based on pairs of measured input/output data (Ljung, 1998). System identification has been extensively studied using statistical methods in the past decade, especially for linear dynamic systems.

  Many approaches have been developed for linear system identification in both the time domain and frequency domain (Ljung

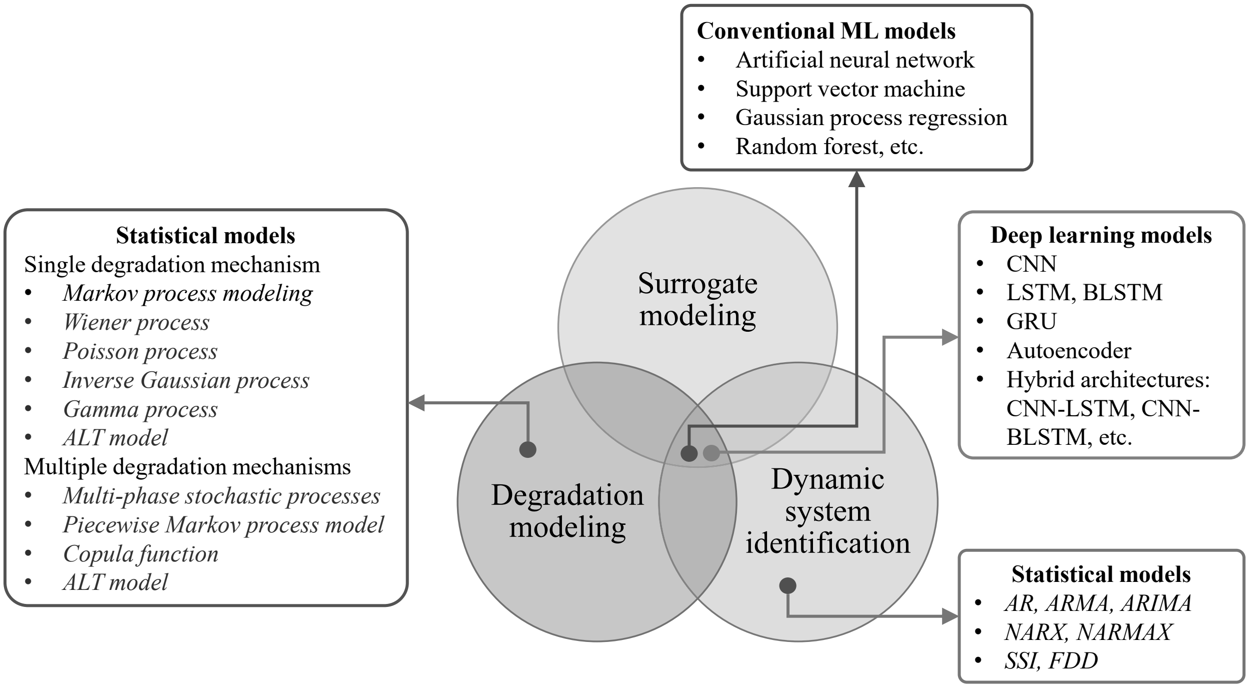


Fig. 8: Overview of data-driven models, where *italic* font represents statistical-modeling methods; and *non-italic* font indicates ML methods.

and Glover, 1981). In the time domain, simple models, like auto-regressive moving average models (ARMA), have been studied for the identification of dynamic systems with stationary responses (Rojo-Álvarez et al., 2004; Martínez-Ramón et al., 2006). To overcome the stationary limitation of ARMA, auto-regressive *integrated* moving average (ARIMA) models have been developed for non-stationary dynamic systems (Xuemei et al., 2010). We highlight the *I* in ARIMA as it denotes the model's ability to model trending data as a series of differences, making it a much more versatile statistical model. Other widely used approaches for time-domain system identification include stochastic subspace identification (SSI) methods (Peeters and De Roeck, 1999) and eigensystem realization algorithms (Juang and Pappa, 1985). In the frequency domain, commonly used modeling methods include frequency domain decomposition (FDD) (Brincker et al., 2001) and the least squares complex frequency domain method (Guillaume et al., 2003). Among the mentioned linear system time-domain and frequency-domain methods, SSI, a parametric identification method based on stochastic state-space models, is one of the most widely used approaches. Many variants of the SSI method have been proposed in the past decade, such as reference-based SSI (Peeters and De Roeck, 1999) and covariance-driven SSI (Döhler et al., 2011). A detailed explanation of SSI is available in (Peeters and De Roeck, 1999; Brincker and Andersen, 2006).

Numerous efforts have also been made to develop statistical methods for nonlinear system identification (Billings, 2013; Piroddi and Spinelli, 2003; Worden et al., 2018). The most widely used methods include the nonlinear autoregressive exogenous (NARX) model (Piroddi and Spinelli, 2003), and the nonlinear autoregressive moving average model with exogenous inputs (NARMAX) model (Billings, 2013). In a NARX model, the value of the current output y_k is represented as a nonlinear function of past values of the output (i.e., $y_{k-1},\ y_{k-2},,\ \cdots, y_{k-p}$) and current and past values of exogenous inputs (i.e., $u_k,\ u_{k-1},\ \cdots,\ u_{k-q}$) as follows

$$\begin{aligned} y_k = g(&y_{k-1},\ y_{k-2,},\ \cdots, y_{k-p}, \\ &u_k,\ u_{k-1},\ \cdots,\ u_{k-q}) + \varepsilon_k, \end{aligned} \tag{2}$$

where $g(\cdot)$ is a nonlinear function, such as a nonlinear polynomial function, p and q are respectively the number of lags of the output and the exogenous inputs included in $g(\cdot)$, and ε_k is a Gaussian random variable with zero mean and unknown standard deviation estimated based on observations along with parameters of $g(\cdot)$. In a NARMAX model, the noise sequences (i.e., $\varepsilon_k, \varepsilon_{k-1}, \cdots, \varepsilon_{k-e}$) are also included in $g(\cdot)$ in addition to past values of exogenous inputs and the response (Billings, 2013). When an ML method is employed to construct the nonlinear function $g(\cdot)$, the nonlinear system identification methods fall into the intersection of statistical methods and ML methods (Worden et al., 2018).

There are also several review papers summarizing various system identification approaches. Namely, Åström and Eykhoff (1971) conducted a detailed review of various system identification methods of dynamic systems, Ljung (2010) shared his perspectives on system identification, Sirca Jr and Adeli (2012) reviewed approaches for system identification in structural engineering, and Kerschen et al. (2006) conducted a comprehensive review on developments in system identification of nonlinear dynamical structures and illustrated them using numerical and experimental examples.

The dynamic system models identified using the system identification methods reviewed above can enable real-time state estimation and control, a highly desirable ability of a digital twin. For instance, He et al. (2019) used a discrete-time state-space model identified from data to model various component-level dynamics in constructing a virtual system in their digital twin model. Similarly, Xia et al. (2021b) used the MATLAB system identification toolbox (see Table 1 in Part 2 of the review paper) to build digital models for their digital twin.

- *Statistical methods for degradation modeling in predictive maintenance (V2P).* One of the most important elements in digital twins is degradation modeling. Degradation modeling is vital for predicting the end of life of a physical system to enable predictive maintenance (i.e., V2P). In some cases, degradation mechanisms of the physical system involve complicated physics which may not be fully understood. In these scenarios, a physics-based degradation model cannot be constructed, and an empirical statistical model must be used instead. Empirical modeling leverages historical data collected from failed units to construct a degradation model. Shown in Fig. 8, many statistical models have been proposed for degradation modeling of both single and multiple failure mechanisms. For example, numerous stochastic process models, including Markov models (Giorgio et al., 2011; Lee et al., 2010; Vega et al., 2021), semi-Markov models (Kharoufeh et al., 2010; Liu et al., 2018a; Compare et al., 2016), Wiener process, Poisson process, inverse Gaussian process, Gamma process, and accelerated life testing (ALT) models (Hu and Mourelatos, 2018), have been developed for degradation modeling of single degradation mechanisms (Ye and Xie, 2015; Gorjian et al., 2010). For degradation modeling with multiple degradation mechanisms, multi-phase stochastic process models (Liao et al., 2021), piece-wise Markov process models (Vega et al., 2020), copula functions (Peng et al., 2016), and system-level ALT models (Moustafa et al., 2021), have been proposed.

While statistical degradation models allow for effective prediction of the end of life of physical systems, there are several challenges that need to be addressed in order for some of the models to be used in a digital twin. For example, a Markov degradation model usually describes the degradation as a set of highly abstract discrete states, such as ratings given in letters (e.g., A, B, C, D). It is often difficult to directly use online sensor data, collected from the physical system continuously and in real-time, to update the highly abstract discrete states. This poses a real challenge in seamlessly establishing the P2V connection in a digital twin system (see Fig. 3). To tackle this challenge, researchers have proposed approaches integrating Bayesian model updating methods reviewed in Sec. 4.2 with Markov chain model for model updating and failure prognostics in Chiachío et al. (2020); Vega et al. (2021). In addition, the uncertainty of a degradation model needs to be properly quantified in order for such models to be fully integrated into a maintenance decision making framework. Accurately quantifying the uncertainty of prognostic models is essential to the effective integration of these models into digital twins for predictive maintenance, especially for high-value physical assets and in safety-critical applications, as will be discussed in Sec. 5.2. A detailed discussion on the role of UQ in digital twins will be discussed in Section 2.1 of Part 2 of the review paper.

3.3.2 ML models

With the increase in IIoT connected devices, big datasets have become commonplace, and advanced data analysis methods have been developed to take advantage of the collected knowledge. ML models, which can learn from a dataset and be used to predict on new, unseen data, have become a popular method for modeling physical systems in a digital twin (Min et al., 2019; Ibrahim et al., 2020; Liu et al., 2020a; Priyanka et al., 2022; Elayan et al., 2021; Ladj et al., 2021; Hu et al., 2021a). As shown in Fig. 8, ML methods can be further categorized into conventional ML methods and newer, emerging DL methods.

Conventional ML methods, as indicated in Fig. 8, include feed-forward neural networks (Chen and Billings, 1992), support vector machines (Zhang et al., 2006), random forests (Chai and Zhao, 2019), and Gaussian process regression (Worden et al., 2018). All of these methods have been used in digital twins for (1) degradation modeling, (2) surrogate modeling to replace first-principles models, and (3) dynamic system identification from data. For example, for surrogate modeling, Chakraborty and Adhikari (2021); Molinaro et al. (2021) used Gaussian process regression models to replace computational simulations in digital twins, and thereby enable real-time prediction; Zhu et al. (2021) trained ML models to build a digital workpiece model for a manufacturing system digital twin; Zhou et al. (2022b) investigated using multiple ML approaches to characterize the relationship between crack size/shape and crack-front stress intensity factors in a helicopter. For the identification of dynamic systems, particularly nonlinear systems, various ML-based approaches have been proposed by following the NARX format given in Eq. (2). When a Gaussian process regression model is used as $g(\cdot)$ in Eq. (2), it is called a GP-NARX model (Doerr et al., 2018). When feed-forward neural networks are used, it is called a NARX-net (Lin et al., 1996).

As a subarea of ML, *DL* is a relatively new modeling technique which uses large multi-connected networks with many parameters to approximate physical process or functions (LeCun et al., 2015). Deep learning models are easily reconfigured to have any number of parameters or architecture, and for this reason, they have a significant advantage over traditional ML models when dealing with massive high-dimensional heterogeneous datasets (Tang et al., 2020; Rassõlkin et al., 2019; Hu et al., 2022a). As shown in Fig. 8 and similar to conventional ML models, DL models have been applied in various digital twin applications for degradation modeling, surrogate modeling, and system identification. In particular, the application of DL models for *degradation modeling* to establish a V2P connection in a digital twin has gained much attention recently. Applications of ML and DL models for degradation modeling in failure prognostics and predictive maintenance will be reviewed and discussed in detail in Secs. 4.4.4 and 5.2. Here, we briefly summarize the application of DL models for surrogate modeling and dynamic system identification.

- In recent years, a large number of DL methods have been proposed for *surrogate modeling*. DL models can replace many expensive physics-based simulations by approximating the underlying physics using the highly-interconnected model parameters. For example, Guo et al. (2022) developed a surrogate model for thermal signature prediction in laser metal deposition using DL models. Tang et al. (2020) proposed a DL-based surrogate modeling method to predict dynamic subsurface flow in channelized geological models. Similarly, Weber et al. (2020) developed a DL surrogate model for earth system simulation models. In deep neural network-based surrogate modeling methods, it is worth highlighting a particular type of neural network – long short-term memory (LSTM) networks, which are a type of recurrent neural network. Unlike regular feed-forward neural networks, LSTM has a feedback loop inside each cell which allows information to persist in time throughout the network (Hochreiter and Schmidhuber, 1997). As a result, it can capture long-term dependencies in time series data. Because of this appealing feature, LSTM networks have been extensively leveraged to build surrogate models for sequence-to-sequence prediction with different number of inputs and outputs (e.g., one to many, many to one, and many to many), such as in aerodynamics (Zhang et al., 2021c), elasto-plastic finite element simulation (Im et al., 2021), and microstructure evolution applications (Montes de Oca Zapiain et al., 2021), among others. Moreover, numerous hybrid DL architectures, such as convolutional neural network (CNN)-LSTM (Zhao et al., 2022) and CNN-Bidirectional LSTM (Zhang and Yin, 2021), have been proposed in recent years for surrogate modeling of various physical systems.

- For *system identification*, DL models are becoming more popular due to their advantage over conventional ML and statistics-based models in handling high-dimensional and heterogeneous data sources, such as videos and images. As mentioned above, the main difference between system identification and surrogate modeling is that system identification is mainly done using sensor data collected during online operation, whereas surrogate modeling may be done with either physics-based simulation data or sensor measurement data.

  There are two main approaches for system identification using DL. In the first approach, DL models are used to learn the *input-output* relationship directly in a nonlinear autoregressive exogenous scheme. In this way, the DL model learns from data to act as a black-box state-space model of the system. The learned states and state transition functions are encoded in the DL model's parameters and are typically not easy to visualize because the model's contain many thousands of parameters. Zhang and Mahadevan (2020) used an LSTM DL model to learn the relationship between the measured aircraft velocity and flight trajectory to facilitate the safety assessment of en-route flight. Ma et al. (2021) used a probabilistic LSTM to learn the relationship between geometric sensor data and thermal elongation. Hu et al. (2022b) developed a grasps generation-and-selection CNN to facilitate performance assessment of robotic grasps in the context of digital twin, where grasping state (i.e., position, rotation angle, and gripper width) was captured and characterized in the form of red–green-blue-depth images. Wu and Jahanshahi (2019) presented a deep convolution neural network-based method for system identification of structural dynamic systems. They showed that CNN-based system identification methods are more robust than conventional multilayer perception neural networks-based methods when the signals are contaminated by large noise.

  In the second approach to system identification using DL, the state-space model of a system is learned in a more easily interpreted, low-dimensional form. The most popular DL model for this is the autoencoder (Masti and Bemporad, 2021; Li and Yang, 2021; Otto and Rowley, 2019). Shown in Fig. 9, an autoencoder is given the name *encoder* because it takes high dimensional inputs I_{k-1} and encodes them as a low-dimensional state vector x_k. The low-dimensional state vector can easily be viewed and visualized since it is of much lower dimensionality than the system inputs and outputs. To estimate the system outputs O_k, the state vector x_k is decoded in the decoder. In a recurrent fashion, the encoded state vector x_k is combined with control system inputs u_k using another small neural network which learns the state transition function $x_{k+1} = f(x_k, u_k)$ to update the state estimate, x_{k+1}, at the next time step. The specifically designed high- and low-dimensional parts of recurrent autoencoder networks offer practitoners the ability to work with high-dimensional inputs and outputs while still maintaining a better level of clarity and visualization by observing the low-dimensional states x. Autoencoders and similar DL models will play a vital role in understanding the many complex input/output relationships in future digital twin frameworks.

We conclude our discussion on data-driven modeling by listing a few topics that are crucial for the successful deployment of data-driven models in a digital twin.

1. **Model selection**: Even though both statistical models and ML models are getting unprecedented attention in recent years, they have their own advantages and disadvantages. For example, statistical models are highly explainable while complex DL models are difficult to

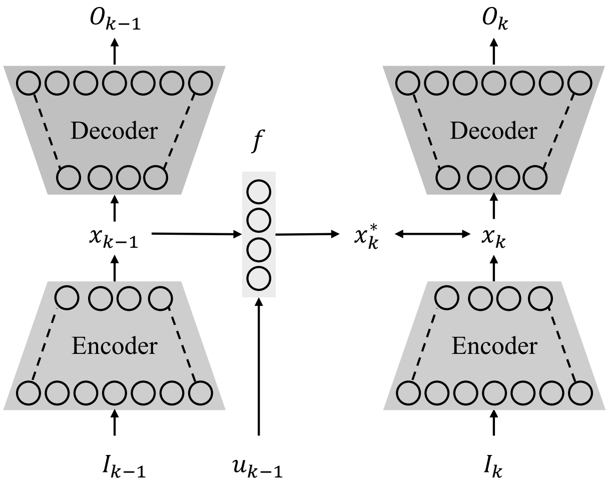


Fig. 9: A recurrent autoencoder-based DL model for system identification of a nonlinear dynamic system (Masti and Bemporad, 2021).

interpret and are often thus used in a black-box way. There are so many data-driven models available, however, selection of an appropriate data-driven model should always follow the "Occam's Razor" principle. For example, if a conventional ML model, such as a Gaussian process regression model, can be employed to build an accurate data-driven model, there is no need to construct a very complicated DL model for the same task. The selection of an appropriate data-driven model for a specific application of digital twin is an important optimization problem.

2. **UQ of ML models**: George E.P. Box famously said "All models are wrong, but some are useful," as recorded in Box (1976). This is also true for all data-driven models including statistical models, conventional ML models, and DL models. It is therefore important to quantify the uncertainty in data-driven models, especially ML models. UQ of ML models allows us to quantify the confidence level of ML predictions. It plays a vital role in determining when the model prediction will go wrong and how bad the prediction is if it is wrong. A detailed discussion of such an important topic is given in Section 2.1.1 in Part 2 of the review paper.
3. **Data collection**: Data is essential for the learning of data-driven ML models. The quantity and quality of available data significantly affect the performance of data-driven models, and eventually the effectiveness of digital twins. In engineering practice, high-fidelity physics-based simulations may be computationally very expensive and physical experiments may be very time- and resource-consuming. In those situations, it is difficult to collect large volumes of high quality training data to build data-driven models. Ways to tackle this challenge include (1) physics-informed ML, reviewed in Sec. 3.4; (2) active learning where data are collected adaptively to refine data-driven models in a data-efficient manner (Cohn et al., 1996), with Bichon et al. (2008) being the first application of active learning to reliability analysis under uncertainty; and (3) online model updating, as will be reviewed in Sec. 4.3, where an ML model is first trained using a small amount of data offline and then is updated using data collected online to improve the prediction accuracy.

3.4 Physics-informed ML

For some aspects of the physics in digital twins, both data from first-principle (or physics-based) simulations and data collected from a physical system may be available. When both physics and data are present, methods presented in Sec. 3.2 and Sec. 3.3 can be integrated, resulting in hybrid physics-ML approaches for modeling physical systems. For instance, Luo et al. (2020) fused a first-principle model with operational data to build a multi-domain model in a digital twin. Similar approaches have been attempted, sometimes labeled as physics-informed ML, for physical system modeling in digital twins (Liu et al., 2018b; Yu et al., 2020; Ritto and Rochinha, 2021; Broo et al., 2022). Incorporating physical knowledge into data-driven ML models can offer several attractive benefits, including better generalization of ML models to unseen "out-of-distribution" test data, as will be discussed in detail in 4.3, the ability to explain the underlying physics of the predictions from the ML models, and speeding up the training process because some physical laws can restrict the feasible region of solutions and limit the values the model parameters can take. According to the way that a first-principle model is integrated with a data-driven ML model, existing hybrid approaches for physical system modeling can be broadly classified into six approaches, shown in Fig. 10 and explained in what follows.

- **Approach 1: Physics-Informed Loss Function** This approach modifies the loss function of a data-driven ML model by adding a physics-informed loss term that penalizes model predictions not compliant with first principles (or physics), thus constraining the training of the ML model toward solutions that comply with physics (Raissi et al., 2019a; Jagtap et al., 2020a; Lu et al., 2021). This approach is the most common way to build hybrid physics-ML models and has found success in many digital twin applications such as solving PDEs and surrogate modeling (Raissi et al., 2019a; Zhu et al., 2019b; Haghighat et al., 2021; Gao et al., 2021; Zhou et al., 2022a), inverse modeling (Raissi et al., 2019b; Jagtap et al., 2020b), fault diagnostics (Shen et al., 2021), and degradation modeling (Yan et al., 2022). This method of physics-informed ML is different than the other approaches in that it does not necessarily require any training data from a physics-based simulation, a significant advantage over other methods.
- **Approach 2: Data Augmentation** This second approach first runs first-principle simulations to generate data at various states an operating conditions of a physical system. Examples of states include healthy, degraded/damaged, and failed for predictive maintenance applications. It then combines the generated data (or synthetic data) with actual data (e.g., experimental data) to create an augmented training dataset for training data-driven ML models. An example that falls into this category is the digital twin model presented by Ritto and Rochinha (2021). The authors generated different damage scenarios using first-principle simulations to augment a training dataset for an ML classifier used for damage detection of a bar structure. Another example is the application of physics-informed ML to degradation diagnostics of lithium-ion batteries, where a simple physics-based half-cell model was used to generate high-degradation simulated data to allow training ML models on small, early-life experimental data (Thelen et al., 2021). Using physics-based synthetic data to train an ML model partially constrains the ML model's output with the physics carried by the first-principle model, thereby improving the generalization performance of the ML model. Additionally, this data augmentation approach can reduce the data size required for training large-scale DL models. A necessary condition for the success of this approach is that the first-principle model captures physics highly relevant to the problem at hand. However, the model does not need to be highly accurate.
- **Approach 3: Transfer Learning** This physics-informed ML approach is inspired by the concept of transfer learning, which has been explored in various digital twin applications, including multi-fidelity modeling, such as Huang et al. (2022b) for melt pool modeling in additive manufacturing, process quality monitoring, such as Kapusuzoglu and Mahadevan (2020) (the third type of strategy) for in-line quality monitoring of additive manufacturing processes, and data-driven prognostics such as Zhang et al. (2018) for turbofan engine prognostics, Shen et al. (2020) for battery capacity estimation, and Wang et al. (2021b) for bearing fault diagnosis. It first pre-trains an ML model using a large quantity of data from first-principle simulations. It then fine-tunes the model using a small quantity of real data (e.g., data from physical system experiments). Similar to the data augmentation approach, this transfer learning approach can improve the generalization of ML models and alleviate the data size requirement. Its success also highly depends on how relevant the physics captured by the first-principle model is to the classification/regression problem that needs to be solved.
- **Approach 4: Delta Learning (Missing Physics)** The basic idea of this delta learning approach is to add a data-driven ML model, co-existing with a first-principle simulation model, that learns to recover the unmodeled physics. This hybrid modeling approach is commonly used in multi-fidelity modeling and quantification of simulation model discrepancy. For example, Jiang et al. (2022) added a polynomial chaos expansion model on top of a simplified physics-based degradation model for failure prognostics of a structural system

called miter gates. The role of the polynomial chaos expansion model was to represent the discrepancy of the simplified physics-based degradation model due to unmodeled dynamics. Yucesan and Viana (2020) built a data-driven LSTM model to augment a physics-based fatigue model, which created a hybrid model for fatigue prognostics of wind turbines.

- **Approach 5: Delta Learning (ML Prediction)** In this approach, an ML model learns residuals on top of initial predictions by another ML model, trained using data generated by a physics-based model. The final predictions are the sum of the initial predictions and residuals. The initial predictions can also be made directly by a physics-based model (Zeng et al., 2020). The logic behind this hybrid physics-ML approach is two-fold: (1) the first-principle model use baseline physics to produce initial, population-based predictions that may not be accurate for individual system units but can generalize well to different operating conditions; and (2) adding the data-driven residuals to compensate for missing physics and unit-to-unit variability. The resulting performance gains are improved generalization over pure data-driven ML and improved prediction accuracy over purely physics-based modeling. Just like the above physics-informed ML approaches, this delta learning approach may not work well if the first-principle model does not capture relevant physics. Issues may also arise when the first-principle model cannot capture the effects of testing or operating conditions on experimental data.
- **Approach 6: ML-Assisted Prediction** This sixth approach to physics-informed ML uses a data-driven ML model to predict the input $\mathbf{x}$ or parameters $\boldsymbol{\theta}$ of a first-principle model. When implementing this approach in practice, the first step is identifying the input or a subset of the parameters as variables, $\mathbf{x}_{\text{FPM}}$ or $\boldsymbol{\theta}_{\text{FPM}}$, to be predicted by the ML model. Then, the ML model estimates or, in some cases, forecasts $\mathbf{x}_{\text{FPM}}$ or $\boldsymbol{\theta}_{\text{FPM}}$. Finally, the first-principle model uses the ML estimates or forecasts to predict the response of interest y at current (estimation) or future (forecasting) time steps. A clear benefit of this approach is the physical interpretability, coming from (1) using physically meaningful variables as intermediate y, predicted by an ML model, and (2) using a first-principle model to predict the final y. Another advantage is the improved generalization performance, attributable to the use of the first-principle model. A typical challenge with this approach is the low identifiability of the first-principle model's input or parameters from observational data. In extreme cases, large errors in the ML model's estimates or forecasts propagate through the first-principle model, causing the final prediction performance to be worse than pure ML. Examples of Approach 6 can be found in grease degradation forecasting (ML) for bearing fatigue life prediction (first-principle modeling) (Yucesan and Viana, 2022) and degradation parameter forecasting (ML) for battery capacity fade prediction (first-principle modeling) Ramadesigan et al. (2011); Downey et al. (2019); Lui et al. (2021).

We note that the above list is not exhaustive but rather an incomplete collection of representative approaches to physics-informed ML. For example, an approach that could, in general, be categorized as Approach 6 is that the output and parameters of a first-principle model are used to augment the input of an ML model, as demonstrated for in-line quality monitoring of additive manufacturing processes in (Kapusuzoglu and Mahadevan, 2020) (the second type of strategy), and turbofan engine diagnostics in Chao et al. (2019) and prognostics in Chao et al. (2022). The only difference is that the information flow is in the opposite direction (i.e., the first-principle model now feeds an additional input to the ML model). A seventh approach could be defining physics-based kernel functions for ML models, such as the FEA-Net (Yao et al., 2020) and physics-informed kernel functions for Gaussian process regression (Yang et al., 2018). Interested readers can find more comprehensive and detailed reviews in review papers more dedicated to the topic of physics-informed ML, such as Karniadakis et al. (2021) for computational physics, Aykol et al. (2021) for battery lifetime prediction, and Willard et al. (2020) for a comprehensive collection of engineering applications. Despite tremendous advances in combining physics and data for physical system modeling, much more research

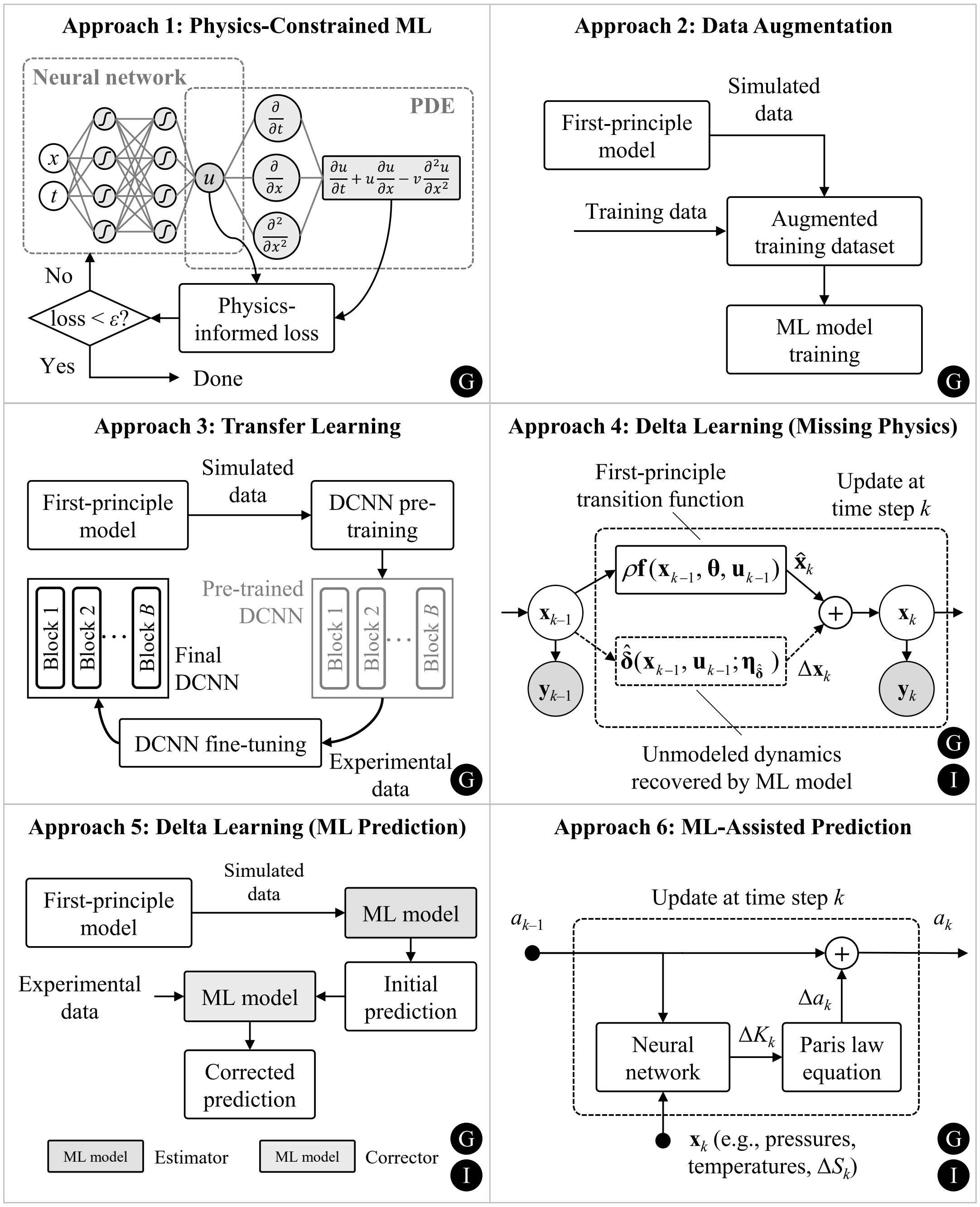

Approach 1: Physics-Constrained ML
Neural network
PDE
x
t
u
$\frac{\partial}{\partial t}$
$\frac{\partial}{\partial x}$
$\frac{\partial^2}{\partial x^2}$
$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2}$
No
loss < ε?
Physics-informed loss
Yes
Done
G
Approach 2: Data Augmentation
Simulated data
First-principle model
Training data
Augmented training dataset
ML model training
G
Approach 3: Transfer Learning
Simulated data
First-principle model
DCNN pre-training
Pre-trained DCNN
Block 1
Block 2
Block B
Final DCNN
DCNN fine-tuning
Experimental data
G
Approach 4: Delta Learning (Missing Physics)
First-principle transition function
Update at time step k
$\rho\mathbf{f}(\mathbf{x}_{k-1}, \boldsymbol{\theta}, \mathbf{u}_{k-1})$
$\hat{\mathbf{x}}_k$
$\mathbf{x}_{k-1}$
$\mathbf{x}_k$
$\hat{\boldsymbol{\delta}}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}; \boldsymbol{\eta}_{\hat{\delta}})$
$\Delta\mathbf{x}_k$
$\mathbf{y}_{k-1}$
$\mathbf{y}_k$
Unmodeled dynamics recovered by ML model
G
I
Approach 5: Delta Learning (ML Prediction)
First-principle model
Simulated data
ML model
Initial prediction
Experimental data
ML model
Corrected prediction
ML model Estimator
ML model Corrector
G
I
Approach 6: ML-Assisted Prediction
Update at time step k
a_{k-1}
a_k
Δa_k
Neural network
ΔK_k
Paris law equation
$\mathbf{x}_k$ (e.g., pressures, temperatures, ΔS_k)
G
I

Fig. 10: Six approaches to construct a hybrid of a physics-based and data-driven ML model: (1) Modifying the loss function of an ML model by adding a physics-informed penalty term (Karniadakis et al., 2021); (2) Generating synthetic data using first-principle simulations (Ritto and Rochinha, 2021); (3) Pre-training an ML model on physics-based synthetic data and then fine-tuning it on experimental data (Shen et al., 2020; Kapusuzoglu and Mahadevan, 2020; Huang et al., 2022b); (4) Correcting a first-principle model by learning unmodeled physics from data (Jiang et al., 2022); (5) Correcting an ML model, trained with physics-based synthetic data, by learning its prediction residual from experimental data (Thelen et al., 2022a); and (6) Learning to predict the input of a first-principle model (Yucesan and Viana, 2022). The icon with "G" inside a black circle indicates improvement in generalization over pure ML; the icon with "I" inside indicates improvement in interpretability.

is needed to address practical challenges impeding industry-scale adoption of hybrid physics-ML modeling approaches. Among those challenges are (1) a lack of large, realistic datasets relevant to different digital twin applications for training, validating, and testing hybrid models and benchmarking their generalization and interpretation performance (see also the discussion in Section 4.2 of Part 2 of the review paper), (2) a shortage of digital twin benchmark problems that allow researchers and practitioners to compare prediction accuracy and computational efficiency of different hybrid approaches and determine their suitability for real-time physical system modeling, and (3) a lack of academia-industry collaborative effort in building open-access, well-maintained platforms that can streamline data sharing, model building, and benchmarking, such as many existing efforts to streamline ML workflows.

3.5 System modeling

A physical system usually consists of many subsystems and components that interact with each other in a variety of means. It is, therefore, important to model the high-level interactions and the overall interconnection structure among different components in the physical system. This section briefly reviews two widely used techniques for modeling interactions in physical systems.

(a) Unified Modeling Language (UML) and Systems Modeling Language (SysML)

Towards systems modeling, the Unified Modeling Language (UML) is a general-purpose language that is commonly used for software development in model-based systems engineering (MBSE) (Ramos et al., 2011). In brief, UML is a graphical modeling language that provides a standardized way to specify the structure of a software system and visualize the interactions of different models in the system (Glatt et al., 2021; Weilkiens, 2011). In engineered systems, UML has been increasingly utilized to characterize the design of a system with regard to its high-level structure, behavior, and interactions. To this end, different diagrams are created to reflect the model in line with the intended purpose from a certain perspective, and these diagrams can be grouped into two categories:

1. **Structure diagram**: It describes the static structure of elements in a system, which is often adopted to document the architecture of software systems. Structure diagram includes class diagram, component diagram, object diagram, deployment diagram, etc. Take the class diagram as an example, it is often used to describe a set of classes, interfaces, and collaborations and their relationships in object-oriented software systems. For example, both classes students and professors belong to a super class person, and the relationship between professors and students is that professors supervise students.
2. **Behavior diagram**: Differing from the structure diagram, behavior diagram emphasizes on characterizing and visualizing the dynamic aspects of the system. It is composed of activity diagram, use case diagram, sequence diagram, state chart. Take the same students versus professors as an example, in the case of course taking, the sequence diagram illustrates the very basic steps in this task ordered by the timeline, starting from the very beginning course registration to homework assignment/submission and homework grading and report. In short, the sequence diagram summarizes the dynamic interactions between different users (i.e., students, professors, administrators) and applications (i.e., course registration, homework submission, homework grading and report) over time.

The promise of UML in modeling complex system interactions has been demonstrated in multiple studies. For example, Luo et al. (2019) leveraged UML to characterize the connections among multiple physical subsystems (i.e., mechanical, electrical, hydraulic) in the digital twin of a CNC machine tool. Note that UML is intensively used for interaction modeling, focusing heavily on the software side.

Differing from UML, the Systems Modeling Language (SysML) is independent of the specific discipline (e.g., software, hardware, personnel, or facilities), and it is a general-purpose architecture modeling language in MBSE through extending and omitting certain UML elements (Weilkiens, 2011). As driven by user requirements, SysML consists of four key pillars: structure (e.g., system hierarchies), behavior (e.g., state machine), properties (e.g., attributes of time variables), and requirements (e.g., requirements' hierarchies and traceability). Using SysML facilitates the design, specification, analysis, verification, and validation of engineering products and processes across a wide range of systems and systems-of-systems. Thanks to SysML's rich features in cross-system extensibility and traceability, there is an increasing number of studies investigating how SysML is adopted in digital twins (Sun et al., 2020b). For example, Bachelor et al. (2019) leveraged the traceability of the allocation of requirements to system function definitions in SysML, and used a digital twin model of an ice protection system to deal with changes in system design through data traceability across the abstracted lifecycle. Zhang et al. (2022a) used SysML as a model fusion tool to establish the relationship among various domain models in a shop-floor digital twin. A similar approach is also developed by Liu et al. (2021a) to model the interaction of different models in Unity3D for their shop-floor digital twin model.

In digital twins, as software elements (i.e., simulation models, database, machine learning models, decision support modules) play increasingly important roles, UML and SysML have been frequently used to manage the growing software complexity, where a unified interface is provided to describe the structure, behavior, and interactions among software modules. Such architecture-level view of all elements and their interdependencies in the overall system brings multiple benefits, such as ensuring the planned functionality and the envisioned benefits, preventing delays of the actual implementation due to planning errors, easing project transfer due to employee leave. The state-of-the-art literature has witnessed prominent adoptions of UML/SysML in the manufacturing industry. A representative usage of UML is showcased by Glatt et al. (2021), where they considered the digital twin of cyber-physical production systems. As the digital system contained a significant number of elements and complex relationships between those elements, Glatt et al. (2021) adopted a UML-based approach to model system structure and component interactions associated with different system functions (i.e., monitoring, diagnostics, prediction). The unified representation of class relationships and system workflow substantially eases the understanding of the object-oriented software systems in digital twins.

(b) Ontology

In computer science and information science, ontology is a prevailing knowledge engineering and representation paradigm to describe the properties of a subject area and define how they are related to each other with explicit descriptions and specifications. Towards this end, ontology encompasses a set of domain-specific concepts, entities, and categories as well as their definitions, attributes, and interrelationships to represent the subject matter of concern. Suppose we are interested in building an ontology map for an IoT system, and we have the basic information related to each device and its digital counterpart. Then a tentative ontology map can be constructed like Fig. 11. In this figure, rectangles denote the distinct types of entities (i.e., device, sensor, and attribute), and arrows represent the relationships between entities (e.g., each sensor has a sensing device).

Note that ontology maps are generalized data models in the sense that they only account for general types of entities and their abstractions with certain properties in common. But they do not include any specific information about an entity in the domain. For example, rather than focus on the characteristics of a specific device, an ontology map concentrates on the general concept of devices by capturing common characteristics that most devices have. The construction of such a unifying conceptual framework significantly fosters the communication and sharing of the structure of information specific to each domain that is understandable by both humans and machines (Castells et al., 2006; Compton et al., 2012). The advantage of ontology in enabling the sharing and reuse of domain knowledge has been one of the key drivers in the recent surge in ontology applications, which has been broadly used in machine learning and information science as a popular means of knowledge representation (Ji et al., 2021). For example, Akroyd et al. (2021) demonstrated a digital twin built with a general-purpose dynamic knowledge graph in the World Avatar project, where ontology-based knowledge representation was repeatedly exploited to facilitate a unifiable interface for data queries.

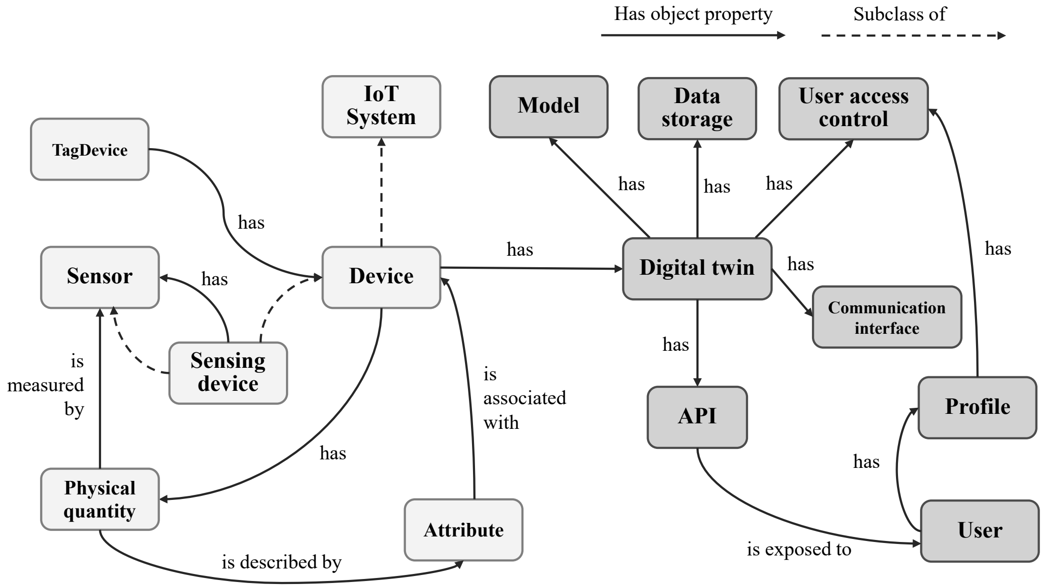


Fig. 11: A simple ontology model for digital twins. Adopted from Steinmetz et al. (2018).

Knowledge graphs, being a popular way to instantiate ontology, are commonly used to connect concepts in ontologies with specific data for knowledge engineering and information retrieval (Wang et al., 2017; Lin et al., 2015). In a knowledge graph, nodes and edges represent the instances of concepts and the relationships between instantiated entities, respectively. Attributable to knowledge graphs' power in querying, inferring, and generating knowledge in the semantic space, they have been adopted as a key enabling technology to achieve semantic interoperability of heterogeneous data and information in engineered systems (Noy et al., 2019; Psarommatis, 2021). For example, Banerjee et al. (2017) exploited the reasoning power in a knowledge graph and developed a query language to extract and infer knowledge from large-scale production line data to support manufacturing process management in digital twins.

In the context of digital twin, ontology and knowledge graphs offer a common language to uniquely refer to each possible object (i.e., component or sensor) in the physical asset. Doing this greatly enhances our flexibility in describing the evolution of digital twin when it is subject to systems/subsystems/components changes (e.g., retrofitting) or modifications in the asset management process (e.g., monitoring) over the entire life-cycle. For example, Erkoyuncu et al. (2020) demonstrated the benefits of an ontology-based framework explicitly, where ontology was used to enable the co-evolution of the digital twin with its physical asset.

4 Physical-to-virtual (P2V) twinning enabling technologies

In this section, we summarize five categories of widely employed physical-to-virtual twinning methods. The P2V twinning methods vary depending on the modeling approach used. Fig. 12 presents the relationship between the different modeling methods reviewed in Sec. 3, the P2V methods reviewed in this section, and the V2P methods that will be discussed in Sec. 5. Shown in the Fig. 12, some of the P2V twinning methods (e.g., probabilistic model updating) are widely used across many different modeling methods, while some other P2V methods (e.g., ontology-based reasoning) are designed for specific modeling techniques. Next, we will explain the five P2V twinning techniques in detail.

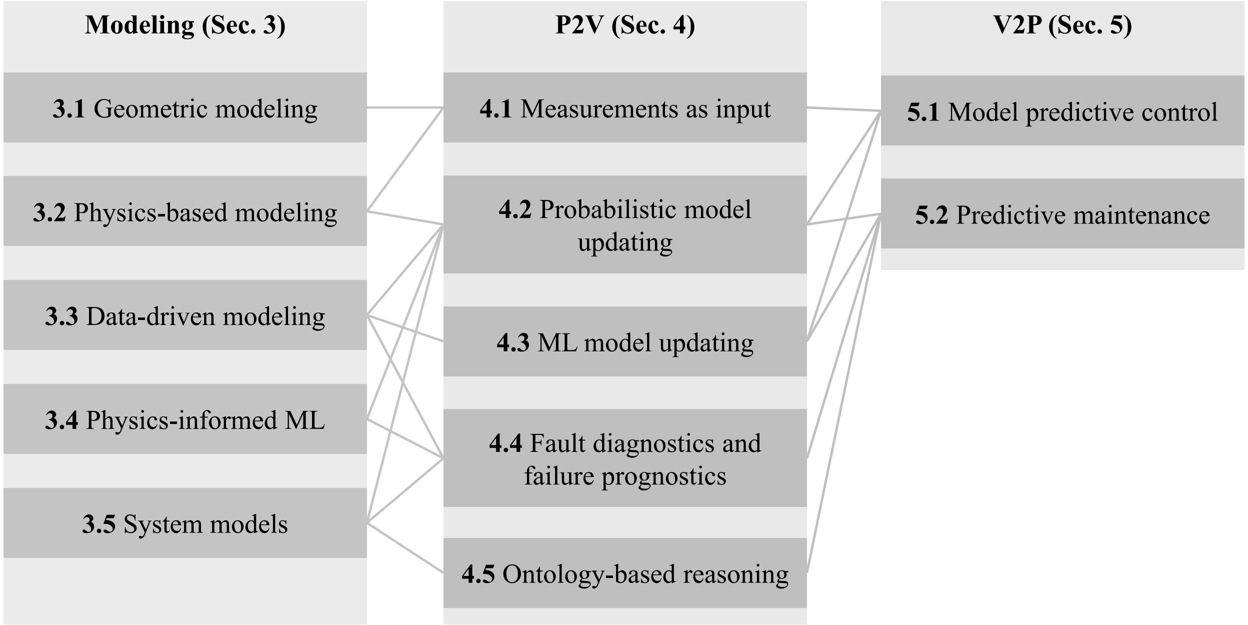

Modeling (Sec. 3)
3.1 Geometric modeling
3.2 Physics-based modeling
3.3 Data-driven modeling
3.4 Physics-informed ML
3.5 System models
P2V (Sec. 4)
4.1 Measurements as input
4.2 Probabilistic model updating
4.3 ML model updating
4.4 Fault diagnostics and failure prognostics
4.5 Ontology-based reasoning
V2P (Sec. 5)
5.1 Model predictive control
5.2 Predictive maintenance

Fig. 12: Relationship between modeling methods, P2V methods, and V2P methods.

4.1 Physical measurements as input to the virtual space

A straightforward way of connecting the physical space with the virtual space (P2V) is to use physical measurements as inputs (Lydon et al., 2019). Doing so enables the digital model in the virtual space to predict the response of physical system in real-time, thus supporting timely risk assessment and decision making (V2P). The current approaches for this type of P2V connection can be categorized into two main groups as follows:

1. *Using physical measurements to update the movements or position of digital models.* This group of P2V updating strategy is mainly for the modeling methods described in Sec. 3.1. For instance, in the manufacturing domain, real-time position data of various components in a manufacturing system collected by the IoT system have been used to generate behavior models in a digital twin (Ricondo et al., 2021), detect disturbances by comparing the position measurements with prediction (Glatt et al., 2021), drive the movement of digital models on the virtual shop floor and thereby optimize manufacturing system configurations or shop scheduling (Leng et al., 2020; Liu et al., 2021a; Fang et al., 2019; Guo et al., 2021; Zhang et al., 2020a, 2021b). In the smart city domain, data on mobility (i.e., movement data of human, vehicles, and other objects), infrastructure, and building conditions are collected in real-time, and are used as inputs for simulations in the virtual space for digital twin-based city development and planning (White et al., 2021; Schrotter and Hürzeler, 2020).
2. *Using physical system monitoring data as input for physics-based analysis or design.* This group of updating strategy is mainly for the modeling methods summarized in Sec. 3.2. For example, in the aerospace domain, Millwater et al. (2019) and Guivarch et al. (2019) employed the physical flight condition data collected from sensors as inputs of a digital model for risk assessment of aircraft structures. In the design domain, Bellalouna (2021) utilized operational data collected from sensors and transmitted the data to a digital environment in the PTC Creo platform for physics-based topology optimization, demonstrating a digital twin concept. In the manufacturing domain, Liu et al. (2021c) performed physics-based hardware-in-the-loop simulation using real-time data from a physical system to replace physical tests, thereby reducing the verification time and cost.

When discussing P2V connections using sensor measurements, it is imperative to also consider the available methods of data transfer, as they should be carefully selected for the application at hand. Digital twin simulations can require substantial speeds and rates of data transfer in order to enable real-time optimization and control (see Section 5.3). In general, data transfer methods can be categorized into two classes: wired and wireless. Wired connections include Ethernet cables (twisted pair cables), coaxial cables, and optical fiber cables, where each cable is rated for different data transfer distances and rates. Wired data transfer is generally preferred, as it is easier to prevent data loss and is generally quicker as the connections are direct from source to receiver. Wireless communication methods include Bluetooth, Wi-Fi, ultra-wide band, and near-field communication (NFC) (Hu et al., 2021b). More recently, the 5G wireless communication standard has come to market offering higher data throughput with lower latency. It is envisioned that 5G technology will enable deployment of more complex and expansive sensor networks to collect and transmit data in real-time for use in digital twins (Cheng et al., 2018). These connected sensor and machinery networks, known more commonly as the industrial internet of things (IIoT), will be responsible for the data collection and transmission that will enable digital twins. Ultimately, each of the aforementioned data transmission techniques will play a critical role in enabling real-time simulation, process control, and optimization for digital twins.

This straightforward P2V connection is only applicable to cases where digital states (e.g., position, mobility, traffic flow) can be directly updated using monitoring data. However, for many engineering problems, digital states cannot be directly updated and are affected by various uncertainty sources. In those cases, more advanced P2V twinning methods are required and will be discussed in the subsequent sections.

4.2 Probabilistic model updating

4.2.1 Introduction to digital state

As shown in Fig. 3, updating a digital system with a time-varying state is an essential characteristic of a digital twin. This state can be called a digital state, defined as a set of digital state variables that characterize the digital model(s) in a digital twin. Examples of digital states enabling predictive maintenance include the health of rotating equipment (e.g., motors, pumps, and compressors) (Wang et al., 2019) and machine tools (Luo et al., 2020) on a production floor, the material properties and structural health parameters of an unmanned aerial vehicle (UAV) (Kapteyn et al., 2020), and the state of charge (SOC) and state of health parameters (e.g., capacity and internal resistance) of a lithium-ion battery cell (Plett, 2004; Hu et al., 2012; Li et al., 2020b). A digital state often cannot be directly measured (or observed) but can be estimated through noisy measurements that depend on the digital state. The digital state of a physical system in operation changes over time and can be estimated as new information about the physical system becomes available. The time evolution of the digital state can be modeled as a dynamic system perturbed by a certain process noise.

Before we talk about the specifics of probabilistic model updating, let us look at a few examples of digital states. A recent paper demonstrated a structural digital twin of a UAV and provided a distinction between a physical state and a digital state (Kapteyn et al., 2021). A key argument in this distinction is that the physical state space should allow capturing variation in the physical system's state (e.g., the structural state of the UAV) and, therefore, could have high complexity and dimensions, while the digital state space should be designed to be simple enough to make the online estimation of the digital state feasible and tractable. As a result, digital models composing a digital twin should provide a digital state space that strikes a balance between complexity (variation in the physical asset captured by the digital state space is sufficient for performing meaningful diagnostics, prognostics, and decision making) and tractability (the number of digital state variables is sufficiently small to allow for good observability for a fixed number of sensor types and quantity of sensors). Fig. 13 illustrates the physical and digital states of the structure in a self-aware UAV application. The digital state encompasses geometric parameters, structural parameters (e.g., material properties and structural health parameters), and boundary conditions. Most of these digital state parameters are calibrated offline, and only a small subset of the structural parameters is dynamically updated online with sensor data, likely to ensure good model identifiability (Arendt et al., 2012). It is worth mentioning that the observational data (control inputs) flow from the physical (digital) to digital (physical) space, which shares similarities with the data and control elements in our proposed digital twin model in Fig. 3.

Another example of a digital state is in an industrial application of digital twin for smart manufacturing (Wang et al., 2019). Although not clearly defined, the digital state of a rotor system seems to refer to parameters of a geometric model, parameters of a rotor dynamics model, and material properties. In the online phase, two parameters of rotor dynamics, namely the critical speed and vibration amplitude under unbalance, are dynamically estimated based on vibration data collected from the rotor system. Similar to Kapteyn et al. (2021), a very small number of digital state parameters are updated online, suggesting the importance of identifying a small yet critical set of parameters for online updating.

The third and last example comes from a review paper on battery digital twin (Li et al., 2020b). Similar to the digital state definitions in Wang et al. (2019) and Kapteyn et al. (2021), the digital state of a lithium-ion battery cell/module/pack can be defined as parameters fully characterizing a digital model of the cell/module/pack. Of particular interest to battery state estimation is the real-time SOC and state of health, the estimation of which are essential functions of a battery management system. Since each battery system in the field is equipped with a battery management system actively estimating and controlling battery SOC and state of health, one could thus argue that the digital twin concept is implemented on every battery system.

4.2.2 State estimation and Bayesian filters

Let us consider a sequence of unknown digital state vectors, $\mathbf{x}_{0:k} = \{\mathbf{x}_0,\ \mathbf{x}_1,\ \cdots,\ \mathbf{x}_k\}$, and a sequence of noisy measurement vectors, $\mathbf{y}_{1:k} = \{\mathbf{y}_1,\ \cdots,\ \mathbf{y}_k\}$. We can formulate a discrete-time state-space model that takes the following form at time step k (i.e., time t_k)

$$\begin{aligned} &\text{State transition}: \ \mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1},\ \mathbf{u}_{k-1}) + \boldsymbol{\omega}_k, \\ &\text{Measurement}: \quad \mathbf{y}_k = \mathbf{g}(\mathbf{x}_k) + \mathbf{v}_k, \end{aligned} \tag{3}$$

where $\mathbf{x}_k$ is the vector of state variables at the k^{th} time step, $\mathbf{u}_k$ is the vector of measured exogenous inputs and often control inputs, $\boldsymbol{\omega}_k$ is the vector of process noise variables for the state, $\mathbf{y}_k$ is the vector of system observations (or measurements), $\mathbf{v}_k$ is the vector of measurement noise variables, and $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$ are the state transition and measurement functions, respectively. The state transition equation in Eq. (3) models the time evolution of the state as a dynamic system, perfectly defined by $\mathbf{f}(\cdot)$, perturbed by a process noise $\boldsymbol{\omega}_k$; it can be rewritten as the transition probability density $p(\mathbf{x}_k|\mathbf{x}_{k-1})$. The measurement equation depicts the dependence of the current measurement $\mathbf{y}_k$ on the current state $\mathbf{x}_k$; it can be rewritten as the conditional density of the measurement $\mathbf{y}_k$ given the state $\mathbf{x}_k$, $p(\mathbf{y}_k|\mathbf{x}_k)$. At this point, the state-space model in Eq. (3) can be graphically represented as a hidden Markov model shown in Fig. 14.

The objective of the state estimation task in Eq. (3) and Fig. 14 is to estimate, in a recursive way, the hidden state at the current time step, $\mathbf{x}_k$, from the current and past noisy measurements (Särkkä, 2013). Mathematically, we want to compute the marginal conditional distribution of $\mathbf{x}_k$ given on all available measurements up to the current step (k), $\mathbf{y}_{1:k}$. This marginal conditional of the digital state can be denoted as $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ and be computed using Bayes' theorem as follows (Leser et al., 2020; Vega et al., 2022)

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{y}_{1:k}) &= \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})d\mathbf{x}_k}, \\ &\propto p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1}), \end{aligned} \tag{4}$$

where $\mathbf{y}_{1:k}$ are observations from t_1 to t_k, "$\propto$" stands for "proportional to", and as described earlier, $p(\cdot|\cdot)$ is a conditional probability density function.

The above equation is analytically intractable. Recursive Bayesian filtering approximates the marginal conditional by recursively executing two

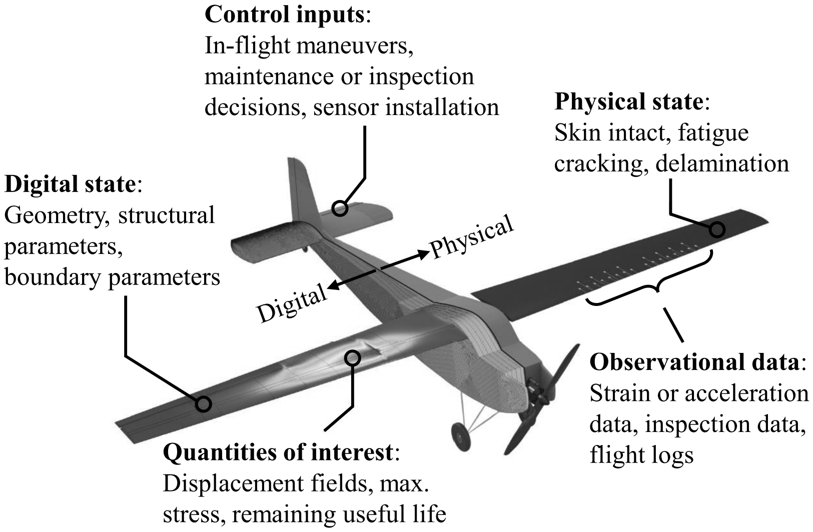


Fig. 13: An abstraction of a digital twin enabling a self-aware UAV, shown as a physical asset-digital twin system. In this example, the digital state could be a small subset of the physical state, allowing for tractable digital-state estimation. Reproduced with permission (Kapteyn et al., 2021). Copyright 2021, Springer Nature.

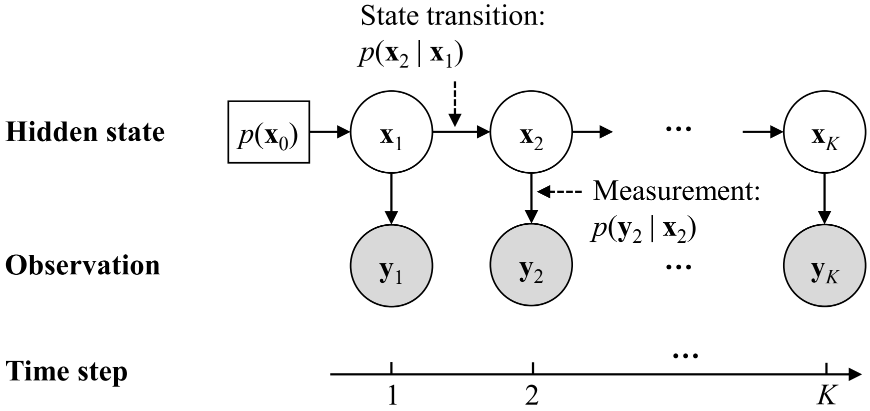


Fig. 14: An illustrative schematic of a hidden Markov model. This model consists of: (1) the hidden state $\mathbf{x}_k$ that evolves according to the state transition equation in Eq. (3), and (2) the measurement $\mathbf{y}_k$ directly related to the hidden state by the measurement equation in Eq. (3). A prior probability density $p(\mathbf{x}_0)$, usually obtained from an offline calibration of the digital model (see Sec. 2.2.2 of Part 2 of the review paper), specifies the initial probability distribution of the hidden state at the 0^{th} time step (i.e., $k = 0$)

steps: a transition step and an update step. Alternative, non-filtering approaches such as Markov chain Monte Carlo and importance sampling approximate the full conditional, $p(\mathbf{x}_{1:k}|\mathbf{y}_{1:k})$. These alternative approaches are not as attractive in the context of digital twin because (1) they could be computationally much more costly than recursive Bayesian filtering, especially when k becomes large, and (2) approximating the full conditional is unnecessary for most digital twin applications.

Table 4 compares four (recursive) Bayesian filters regarding system nonlinearity, state distribution constraints, and computational efficiency (Särkkä, 2013). If the state-space model in Eq. (3) is perfectly linear and Gaussian (i.e., $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$

are linear functions and $\boldsymbol{\omega}_k$ and $\mathbf{v}_k$ follow Gaussian distributions), the Kalman filter, one of the simplest and most efficient Bayesian filters, finds the exact solution to the recursive filtering problem, i.e., the solution to the conditional marginal is exact. Suppose $\mathbf{f}(\cdot)$ or $\mathbf{g}(\cdot)$ is nonlinear and or $\mathbf{v}_k$ is non-Gaussian. In that case, the extended Kalman filter provides a first-order approximation to the marginal posterior by (1) linearizing the state-space model, and (2) assuming Gaussian noise (Gustafsson, 2010). This approximation may be accurate for state-space models whose $\mathbf{f}(\cdot)$ or $\mathbf{g}(\cdot)$ has mild nonlinearity and where true marginal posteriors are unimodal and symmetric. For state-space models with higher function nonlinearity, the unscented Kalman filter propagates a small set of points (called sigma points), sampled based on the prior mean and covariance of $\mathbf{x}_k$, through the nonlinear $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$ functions, from which a posterior mean and covariance of $\mathbf{x}_k$ are estimated while assuming Gaussian. It can accommodate quadratic (second-order) terms in general and even cubic (third-order) terms in some cases when $\boldsymbol{\omega}_k$ and $\mathbf{v}_k$ are Gaussian.

Particle filters differ drastically from the family of Kalman filters in that it does not assume state and noise are Gaussian and characterize uncertainties using Gaussian mean and variance. Instead, particle filters draw a large set of weighted samples (or particles) from the marginal posterior (or a distribution proportional to the marginal posterior) at each time step and use these particles as an approximation to the posterior (Särkkä, 2013; Gustafsson, 2010). An example of pseudocode for a generic particle filter algorithm is given in Sec. 7 (see Figure 28). As shown in the pseudocode, particle filtering consists of three key steps: state transition, weight evaluation and normalization, and resampling. An easy-to-understand explanation of a particle filter can be found in an animated video on YouTube (Svensson, 2022b). This video gives a toy example of positioning an aircraft based on measurements of the distance to the ground and flight elevation. The MATLAB source code for this illustration can be accessed openly on GitHub (Svensson, 2022a). As a result of using the particle approximation, particle filters work well on any non-Gaussian state ($\mathbf{x}_k$) and noise ($\boldsymbol{\omega}_k$ and $\mathbf{v}_k$) distributions and highly nonlinear transition and measurement functions ($\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$). These attractive advantages come with a price: higher computational costs. For example, the unscented Kalman filter, the most computationally complex member in the Kalman family, needs to propagate a set of 7 sigma points to estimate the SOC, capacity, and internal resistance position velocity of a lithium-ion battery (capacity and internal resistance are two state-of-health parameters that affect the energy and power capability of a battery). However, a particle filter may propagate hundreds of weighted particles at each step to achieve good convergence speed and estimation accuracy. Also, particle filters do not scale easily to high-dimensional state spaces, where we may have no particles near the true state, also known as particle deprivation, and the numbers of particles required to achieve reasonable estimation become prohibitively high. Therefore, applications of particle filters for state estimation in the digital twin context have been mostly limited to a small number of state dimensions (typically < 5), consistent with what we found in Li et al. (2020b); Wang et al. (2019); Kapteyn et al. (2021).

It is worth mentioning that approaches have been developed that use cheap-to-evaluate surrogate models to replace the state transition function ($\mathbf{f}(\cdot)$), measurement function ($\mathbf{g}(\cdot)$), or both when these functions are computationally expensive. These approaches alleviate, to some degree, the computational burden imposed by particle filers by allowing a more efficient propagation of thousands or even millions of weighted particles with a lower computational cost. These efforts make particle filter a popular technique for state estimation in many digital twin applications as reported in the literature Li et al. (2017); Kapteyn et al. (2021); Ye et al. (2020).

Two other state estimation problems relevant to digital twin are called smoothing and prediction. These two problems aim to estimate past (smoothing) and future (prediction) states, respectively. Figure 15 illustrates the time requirements for all three state estimation methods. Each of the three methods play a different role in digital twins because they focus on different time frames. Smoothing methods operate on previously collected data and generally reduce the size of the data in the process. For example, smoothing methods can be used to understand a physical system's state history over time after the data has

Table 4: A comparison of four different recursive Bayesian filtering methods

Quantity of interest	**Kalman filter**	**Extended Kalman filter**	**Unscented Kalman filter**	**Particle filter**
Ability to handle nonlinear functions ($\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$)	Linear	Mildly nonlinear (linearized with first- or second-order Taylor expansion)	Nonlinear (can handle quadratic and even cubic terms)	Nonlinear
Process noise and measurement noise ($\boldsymbol{\omega}_k$ and $\mathbf{v}_k$)	Gaussian	Gaussian	Gaussian	Gaussian and non-Gaussian (no limitation)
Posterior state distribution ($p(\mathbf{x}_k\|\mathbf{y}_{1:k})$)	Gaussian	Unimodal (only one peak) and symmetric	Unimodal (only one peak)	Unimodal and multimodal (no limitation)
Computational efficiency	High	Medium	Medium-low	Low

been collected. On the contrary, prediction methods are concerned with estimating future states using historical data. Shown in Fig. 15, the *prediction* state estimation method is estimating the state of the system at time k using historical data. While smoothing was used to understand historical system state, prediction is typically used for understanding future system state. Prediction in digital twin is typically used for planning and control; two use cases which require some degree of understanding about the trajectory of system states.

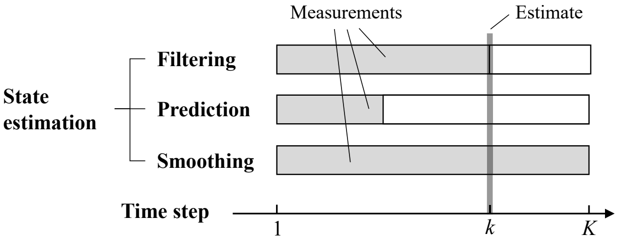


Fig. 15: A schematic distinguishing among three different state estimation problems. The differences lie in the period of the available measurements relative to the state estimation time.

In digital twin applications, state estimation results are used to inform decision making algorithms. The timeliness required for effective decision making using digital twin influences the choice of state estimation methodology. For example, fault diagnosis of machinery requires that the machine's health status be estimated at the present time. In this case, the ideal state estimation methodology to use in a fault diagnosis digital twin would be filtering, as it provides state estimates at the present time. On the other hand, health forecasting and remaining useful life (RUL) prediction require state estimates of system health at future times, and are thus only enabled by prediction methods which predict state at future time steps (See Fig. 15) (Wang et al., 2019; Tian et al., 2011; Bangalore and Patriksson, 2018). Another example is that a structural digital twin of a UAV with an updated damage state can be used to predict (by simulation) the probability of failure for an upcoming flight maneuver and thus informing flight path planning and control (Kapteyn et al., 2021). The idea is to allow a structurally healthy UAV to perform aggressive maneuvers to reduce flight distance and time and constrain a structurally damaged UAV to act more conservatively to incur minimal damage in future maneuvers and prevent unexpected structural failure. The last example is that the state-of-charge and state-of-health estimate of a lithium-ion battery can be used to optimize the current profile (i.e., charging C-rate vs. SOC) during fast charging (Li et al., 2020b, 2021; Wei et al., 2021).

4.2.3 Dynamic Bayesian network

A Bayesian network (BN), which is also called a Bayes net, is a probabilistic graphical model that represents a set of random variables and their probabilistic relationships as a directed acyclic graph (Murphy, 2002; Friedman et al., 1997). A dynamic Bayesian network (DBN) extends a standard BN by considering the time evolution of variables and is used to model dynamic systems. It is a generalization of the hidden Markov model given in Fig. 14 (see Sec. 4.2 (b)). The difference between a hidden Markov model (HMM) and a DBN is that hidden Markov model uses a single hidden state variable to represent the entire state space whereas the DBN represents the hidden state as a set of random variables connected in a graph (Murphy, 2002). A DBN allows for the modeling of nonlinear dynamic systems with arbitrary nonlinearities and distributions.

Fig. 16 presents an illustrative example of a DBN with five nodes. At each time instant t_k, the DBN has a static BN. The static BNs at two adjacent time instants t_k and t_{k+1} are connected through a transition BN. As a directed acyclic graph, a DBN consists of two essential elements, namely vertices (also called nodes) and edges. In a DBN, the vertices or nodes are random variables, such as $x_{1,k}, x_{2,k}, x_{3,k}, x_{4,k},$ and $y_{1,k}$ in Fig. 16. The edges directed from one node to another, such as $x_{3,k} \to y_{1,k}$ and $x_{1,k}, x_{2,k} \to x_{3,k}$, represent the probabilistic causality relationship between the nodes.

Nodes in a DBN could represent discrete random variables or continuous random variables. For instance, as part of the U.S. Air Force funded digital twin effort, Li et al. (2017) represented bolt looseness and a damage state (i.e., elastic/plastic zone) as discrete random variables and a crack length as a continuous random variable in their DBN-based digital twin that modeled the fatigue crack growth on an aircraft wing. Moreover, the nodes that an edge is directed from and directed to are called the parent node and the child node, respectively. For example, for edge $x_{1,k}, x_{2,k} \to x_{3,k}$ in Fig. 16, $x_{1,k}, x_{2,k}$ are parent nodes and $x_{3,k}$ is a child node. Nodes without parent nodes are called root nodes (e.g., $x_{1,k}, x_{2,k}$ in Fig. 16).

Edges connecting parent nodes with the corresponding child node(s) in a DBN, such as $x_{3,k} \to y_{1,k}$ and $x_{4,k} \to y_{1,k}$ in Fig. 16, represent the causality relationship(s) between the parent node(s) and the child node(s). The probabilistic causal relationships between a group of parent nodes connected to a common child node are modeled as a conditional probability table (CPT) for discrete nodes and a conditional probability density (CPD) function for continuous nodes. The CPT contains the conditional probability mass function of a child node covering all possible realizations of its parent nodes. For example, the probability of having a bolt looseness or not for a given realization of load condition of an aircraft is described as a CPT in Li et al. (2017). The CPD could be a simple, analytical probability density function or a numerical probabilistic predictive model, such as a Gaussian process regression model (Karve et al., 2020), a data-driven stochastic process model (Yu et al., 2021), or a reduced-order model with noise (Kapteyn et al., 2021). For instance, if Gaussian process regression is employed to construct the CPD of $x_{3,k}$ given in Fig. 16, a Gaussian process regression model $x_{3,k} = \hat{G}(x_{1,k},\ x_{2,k})$ is first built according to the input-output dependence depicted by the edges $x_{1,k}, x_{2,k} \to x_{3,k}$. After that, we have the CPD of $x_{3,k}$ as

$$p(x_{3,k}|x_{1,k}, x_{2,k}) = \phi\left(\frac{x_{3,k} - \mu_{\hat{G}}(x_{1,k}, x_{2,k})}{\sigma_{\hat{G}}(x_{1,k}, x_{2,k})}\right), \tag{5}$$

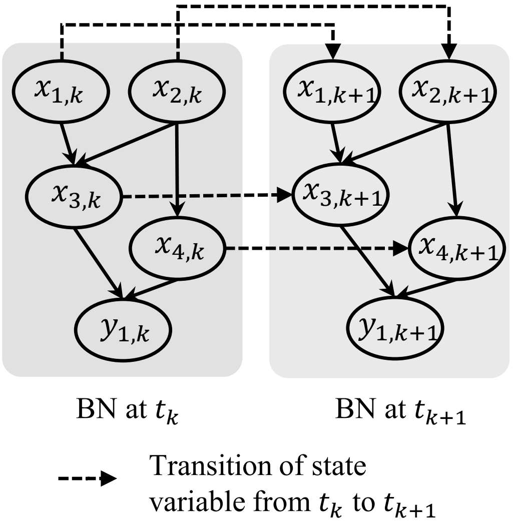


Fig. 16: Illustration of a DBN, which is a generalization of the hidden Markov model given in Fig. 14.

in which $\mu_{\hat{G}}(x_{1,k}, x_{2,k})$ and $\sigma_{\hat{G}}(x_{1,k}, x_{2,k})$ are respectively the mean and standard deviation of the Gaussian process regressor-predicted $x_{3,k}$ for given values of the parent nodes $x_{1,k}$ and $x_{2,k}$, and $\phi(\cdot)$ is the probability density function of a standard normal random variable.

The physic-based models described in Sec. 3.2, such as FEA models, can also be used to construct CPDs by accounting for uncertainty sources in the models. Since digital twins require real-time model updating and each updating run may need to evaluate CPTs or CPDs thousands of times, high-fidelity physics-based models are usually replaced with computationally cheaper data-driven surrogate models (see Sec. 3.3) in the practical implementation of DBNs. In addition to the *nodes* and *edges*, as mentioned above, a transition BN is used to connect two BNs at two adjacent time instants. This is essential to making the BN *dynamic.* Taking the DBN given in Fig. 16 as an example, the transition of each state variable between two adjacent time instants t_k and t_{k+1} is given by

$$p(x_{i,k+1}|x_{i,k}), \forall i = 1,\ 2,\ 3,\ 4, \tag{6}$$

where $p(x_{i,k+1}|x_{i,k})$ is the transition probability of state variable x_i from t_k to t_{k+1}, which is defined by a CPT or CPD and shown as a dashed edge in Fig. 16 (i.e., $x_{i,k} \to x_{i,k+1}, i = 1, 2, 3, 4$).

An important property of BNs and DBNs is the *conditional independence* property, which means that the nodes are independent from each other if they are not connected by directed edges, given the values of their parent nodes. For instance, for given values of $x_{1,k}$ and $x_{2,k}$ in Fig. 16, $x_{3,k}$ and $x_{4,k}$ are independent from each other. Thanks to this property and based on the chain rule of conditional probabilities, the likelihood function of a DBN can be expressed as a product of individual CPTs and/or CPDs conditioned on the parent nodes (see an example given in Appendix B). The decomposition of the joint probability density into a product of individual CPTs and CPDs as illustrated in Eq. (14) in Appendix B really facilitates model updating and the construction of complex DBNs for digital twins. One just needs to properly define the CPTs and CPDs for individual nodes and connect them using a directed acyclic graph. After that, the inference can be performed using generic Bayesian inference methods including the commonly used Markov chain Monte Carlo simulation (Brooks, 1998) and sequential Monte Carlo methods such as particle filtering as described in Sec. 4.2 (b), and given in Appendix A. As has been pointed out by Murphy (2002), the flexibility of DBNs in incorporating various CPTs and CPDs using directed acyclic graphs makes DBNs advantageous over various hidden Markov models including classical hidden Markov models and hierarchical hidden Markov models. The fact that the CPDs in a DBN could be any type of probabilistic model with any nonlinearities and distribution types makes DBNs a flexible tool capable of fusing heterogeneous data sources (e.g., images, text, audio) for model updating (a.k.a. information fusion). Kapteyn et al. (2021) suggested DBNs as the foundation of a unifying mathematical formulation for digital twin at scale, by integrating Bayesian statistics, dynamical systems, and control theory. They also demonstrated the DBN concept using the UAV digital twin example as given in Fig. 13.

The directed acyclic graph in a BN or DBN is usually constructed based on physical knowledge of the causality relationships between different nodes, such as the logical relationships defined by the ontology and SysML as been described in Sec. 3.5. It is possible that the logical or causal relationship is unknown. In that case, the directed acyclic graph/causal relationships could be learned from data using BN learning methods, such as minimal description length, Bayesian–Dirichlet equivalence, and mutual information test (Scutari, 2009; Murphy, 2002; Hu and Mahadevan, 2018b). In BN learning, the most probable topology of the graph is identified by minimizing or maximizing a score function, such as the aforementioned minimal description length, Bayesian–Dirichlet equivalence, or mutual information. A tutorial on BN learning can be found in (Heckerman, 2008). Also, there are several open-sources toolkits available for the inference, learning, and construction of DBNs, such as the Graphical Models Toolkit (Bilmes) and the Bnlearn Python package (Taskesen).

4.2.4 State and parameter estimation

When a digital twin involves multiple types of physics, the time scales of different digital state variables could be quite different. For example, the capacity of a lithium-ion battery cell typically fades slowly and only shows noticeable changes in months or years. In contrast, the cell's SOC typically changes quickly with time and may go from 0% (fully depleted) to 100% (fully charged) within minutes in a fast charging cycle. As will be discussed in the battery case study in Sec. 3 of Part 2 of the review paper, SOC and capacity are two key performance measures that together assist in managing the health of lithium-ion batteries. The state estimation literature usually refers to estimating digital state variables that change very slowly or do not change with time as *parameter estimation*. Estimating digital state variables that change rapidly with time is called *state estimation*. Note that both the hidden state $\mathbf{x}$ and parameters $\boldsymbol{\theta}$ are digital state variables and part of the digital state.

When the state $\mathbf{x}$ is known and noise-free, then state estimation will not be needed, and only parameter estimation is needed. A straightforward way to estimate $\boldsymbol{\theta}$ is to solve a least squares problem, formulated based on the measurement function $g(\cdot)$ for a one-dimensional known state x and measurable response y as follows:

$$\min_{\boldsymbol{\theta}} \sum_{k=1}^{K} (y_k - g(x_k,\ \boldsymbol{\theta}))^2, \tag{7}$$

where K is the total number of measurements available for estimating $\boldsymbol{\theta}$ and $g(\cdot)$ is the measurement function that predicts y at state x, given the vector of model parameters $\boldsymbol{\theta}$. Here, $g(\cdot)$ can be a physics-based model as discussed in Sec. 3.2 or a data-driven model as discussed in Sec. 3.3. Three examples of least squares parameter estimation for a physics-based model are reported in (1) Wang et al. (2019) for estimating two fault-sensitive parameters of a rotor dynamics model for unbalance diagnostics of a rotor system, (2) Ramadesigan et al. (2011) for estimating five degradation-sensitive parameters of an electrochemical model for every charge/discharge cycle for battery capacity fade prediction, and (3) Peng et al. (2020) for estimating five degradation-sensitive parameters of an equivalent circuit model for condition monitoring of a buck DC-DC converter. Least squares estimation can also be used to estimate the parameters of data-driven models, such as estimating the weights and biases of an artificial neural network from a training dataset $\{(x_1, y_1), \ldots, (x_K, y_K)\}$.

If $g(\cdot)$ is linear, Eq. (7) can be solved using linear least squares. When measurement data comes in a stream, the Kalman filter discussed in Sec. 4.2 (b) and recursive least squares can be used to ensure a fixed computational cost at each time step Simon (2006). They produce optimal estimates, and if the assumption of Gaussian measurement noise holds, they are equivalent.

If $g(\cdot)$ is nonlinear, numerical optimization algorithms, such as Gauss-Newton, gradient descent, and Levenberg–Marquardt, and population-based metaheuristic optimization algorithms, such as genetic algorithms and particle swarm optimization, can be used to solve Eq. (7). However, a potential issue is that the resulting estimate of $\boldsymbol{\theta}$ is deterministic. This estimate does not capture prior knowledge on $\boldsymbol{\theta}$ or uncertainty in parameter estimation (e.g., due to low identifiability and measurement noise). This issue can be mitigated by deriving an approximate posterior using a Markov chain Monte Carlo method combined with a prior distribution of $\boldsymbol{\theta}$ under a Bayesian calibration framework (Ramadesigan et al., 2011). Although sequential Bayesian calibration (e.g., a sequential version of a well-known Bayesian calibration framework referred to as the Kennedy and O'Hagan (KOH) framework (Kennedy and O'Hagan, 2001)) has been shown to track time-varying parameters, this approach may be computationally intensive and require longer run times than the response time needed to track sudden changes (Ward et al., 2021). A detailed discussion on the applicability of Bayesian calibration to UQ of dynamic system models is given in Sec. 2.1.2 in Part 2 of the review paper. Sequential, recursive filtering alternatives to these iterative optimizers are nonlinear extensions of the Kalman filter (e.g., the extended Kalman filter and unscented Kalman filter) and particle filters, which are more attractive in computational efficiency. It is not uncommon to see these recursive filtering alternatives succeed in training nonlinear models like neural networks with fixed optimal parameters (Puskorius and Feldkamp, 1994; Aitchison, 2020), with a possible extension to time-varying parameters (Wagner et al., 2021). A new state-space representation can be defined for parameter estimation in the presence of a known and clean state. This new state-space model takes the following form:

$$\begin{aligned}&\text{State transition}: \ \boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \mathbf{r}_k,\\ &\text{Measurement}: \ \mathbf{y}_k = \mathbf{g}(\mathbf{x}_k,\ \boldsymbol{\theta}_k) + \mathbf{v}_k,\end{aligned} \tag{8}$$

Now suppose that g is highly nonlinear, $\boldsymbol{\theta}$ is high-dimensional (e.g., consisting of > 5 parameters), and the computational time and cost are not a concern. In that case, population-based optimization algorithms, such as genetic algorithms and particle swarm optimization, could be better alternatives due to the simple implementation and applicability to a wide variety of g function forms and nonlinearities (Wang et al., 2019; Peng et al., 2020).

When the state is unknown or given with noise, state estimation and parameter estimation are needed. In these cases, state-space models have unknown parameters $\boldsymbol{\theta}$ and hidden states $\mathbf{x}$ that need to be estimated. As mentioned in Sec. 4.2 (a), $\boldsymbol{\theta}$ is part of the digital state, and these parameters may first be calibrated offline but must be updated online. With both a hidden state and unknown parameters, the state-space model in Eq. (3) becomes

$$\begin{aligned}\text{State transition}: \ &\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1},\ \boldsymbol{\theta}_{k-1},\ \mathbf{u}_{k-1}) + \boldsymbol{\omega}_k,\\ &\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \mathbf{r}_k,\\ \text{Measurement}: \ &\mathbf{y}_k = \mathbf{g}(\mathbf{x}_k,\ \boldsymbol{\theta}_k) + \mathbf{v}_k,\end{aligned} \tag{9}$$

where $\boldsymbol{\theta}$ is the vector of unknown model parameters which changes slowly or do not change over time, $\mathbf{r}_k$ is the vector of process noise variables for the parameters at the k^{th} time step.

Similar to when the state is known and clean, iterative optimization algorithms can be used to jointly estimate $\mathbf{x}$ and $\boldsymbol{\theta}$. Sequential approaches such as recursive Bayesian filtering become favorable alternatives for online, close to real-time state estimation with limited computational power (e.g., in an embedded application). A straightforward sequential approach is to combine the state and parameters into an augmented state, $\mathbf{z} = [\mathbf{x};\ \boldsymbol{\theta}]$, and estimate $\mathbf{z}$ using the extended Kalman filter or unscented Kalman filter. This approach is referred to as joint (extended or unscented) Kalman filtering Cox (1964), which was attempted more than half a century ago. As a new measurement from a physical system becomes available, this joint version of the extended Kalman filter simultaneously updates the state and parameter estimates. A drawback reported for this approach is divergence and bias in the parameter estimation Ljung (1979). An alternative is the dual extended or unscented Kalman filter that separately considers state and parameter estimation (Wan and Nelson, 2001). Instead of using fixed state transition and measurement functions such as in Eq. (#), these approaches alternate between estimating the state given the most recent parameter estimate and estimating the model parameters given the most recent state estimate. Two notable applications of the dual extended Kalman filter are battery state (SOC) and parameter (capacity) estimation Plett (2004) and vehicle state (e.g., longitudinal and lateral velocities) and parameter (e.g., vehicle mass and moments of inertia) estimation Wenzel et al. (2006). Compared to the hidden state $\mathbf{x}$, the parameters $\boldsymbol{\theta}$ tend to vary more slowly with time. For example, the SOC and capacity of a lithium-ion battery cell change on two largely different time scales, as mentioned earlier. The different time scales of these digital state variables present unique challenges in estimating these variables. More recently, a multiscale filtering framework (see Fig. 17) was proposed to estimate the SOC and capacity of a lithium-ion battery cell (Hu et al., 2012). Multiscale filtering utilizes the temporal nature of SOC (fast varying) and capacity (slowly varying) to improve the efficiency and accuracy in estimating both the state and parameter over the joint and dual versions of the extended Kalman filtering.

4.3 ML model updating

ML models are an increasingly critical component of digital twins, and they are often deployed in an environment with periodic operational changes in the physical world, such as sensors addition/removal, manufacturing process updates (e.g., add/remove a step in a manufacturing process), changes in measurement device settings/performance, etc. These significant changes might lead to test data that is sufficiently far from the distribution of the data on which the ML model is trained. Such phenomenon is typically referred to as “out-of-distribution” in the ML field (Lee et al., 2018; Zhang et al., 2022b). An ML

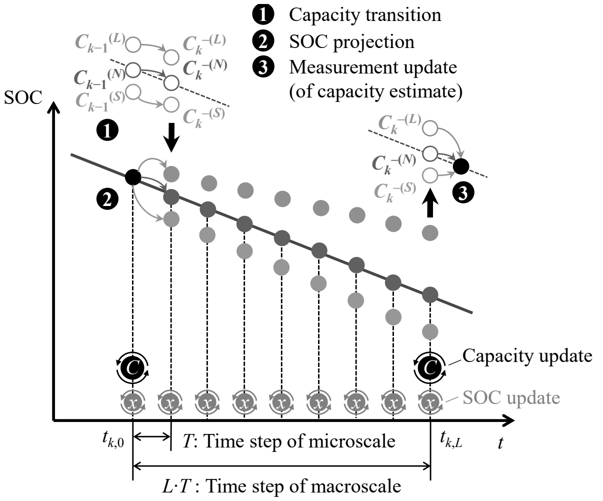


Fig. 17: Illustration of multiscale extended Kalman filtering for battery SOC and capacity estimation (adapted from Hu et al. (2012)).

model is often trained with a closed-world assumption (i.e., the training data and the test data are assumed to follow the same distribution). As a result, if out-of-distribution test data is not incorporated in model training timely and properly, ML model performance will drift significantly.

It is crucial to detect the out-of-distribution situation and update the ML model to account for the operational changes associated with the physical asset. ML model updating will prevent severe performance degradation of a trained and deployed ML model. Updating digital models based on the continuous monitoring of the physical counterpart is an important characteristic of digital twins (Chakraborty et al., 2021). Besides, the timely update of the ML model is also the key to ensuring a consistent and acceptable predictive performance in characterizing the trend of quantities of interest associated with physical assets. One of the most commonly adopted paradigms is to utilize a performance-based indicator to determine when model drift onsets and subsequently trigger an update of the deployed ML model. ML model updating can be implemented in the cloud or on the edge. In the ML community, model drift (also referred to as model decay) is developed as an umbrella term to characterize the degradation of a model's predictive power due to unforeseeable changes in the environment. In practice, model drift often manifests itself in two principal forms:

1. **Data drift** (also referred to as covariate drift): As mentioned before, the training of ML models is established upon the assumption that the training data distribution is representative of the test data distribution. However, this assumption can easily get violated in many real-world applications (Sugiyama et al., 2008). The term " data drift" is used to describe the situation that statistical properties of the production data differ substantially from those of the training data. and it can be mathematically represented as $p^{\text{train}}(\mathcal{X}) \neq p^{\text{test}}(\mathcal{X})$.

   Fig. 18 showcases an instance of data drift, where the relationship between $\mathcal{X}$ and $\mathcal{Y}$ remains the same (see the target function in Fig. 18). Unfortunately, $\mathcal{X}$ follows two different distributions in the training data and test data. As a result of data drift, the function learned from the training data leads to erroneous predictions on the test data.
2. **Concept drift**: This form of model drift characterizes the phenomenon in which the underlying relationship between $\mathcal{X}$ and $\mathcal{Y}$ changes in non-stationary environments (Lu et al., 2018). In essence, the quantity we are trying to predict evolves over time. Mathematically, concept drift is represented as $p^{\text{train}}(\mathcal{Y}|\mathcal{X}) \neq p^{\text{test}}(\mathcal{Y}|\mathcal{X})$.

   Fig. 19 demonstrates an example of concept shift using the same regression problem. Obviously, the underlying functional relationship between $\mathcal{X}$ and $\mathcal{Y}$ has substantially changed from the training data to test data.

In either case, model drift poses significant robustness and reliability challenges in practical ML applications. Such drift has led to substantial ML model performance degradation (Tsymbal, 2004). Safeguarding an ML model in digital twins from this issue entails a close monitoring of ML performance variation using a drift indicator and proactive update of the ML model on an as-needed basis. Here, a drift indicator is typically a quantitative application-specific metric. For example, one approach is to monitor an ML model's test data for drift from the training data with a statistic metric (e.g., Kullback–Leibler divergence or Jensen-Shannon divergence). A second approach is to monitor the variation of model predictive quality over a time window, allowing trending model quality. A third approach is to calculate a

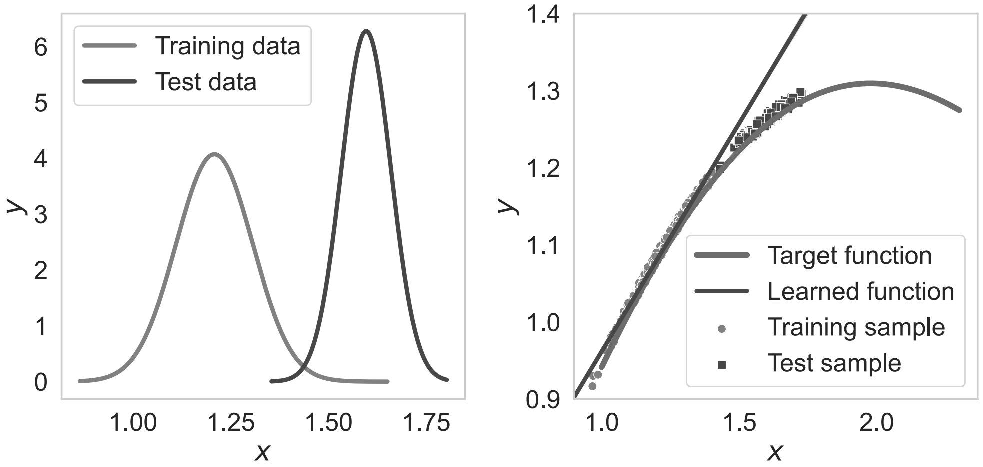


Fig. 18: Demonstration of data drift. Adopted from Zhang et al. (2022c)

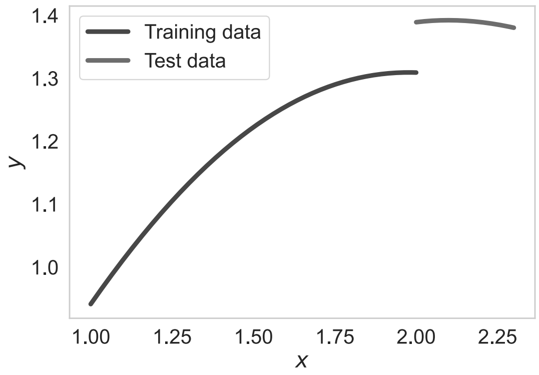


Fig. 19: Demonstration of concept drift

mean error between the ML predictions and actual measurements in the recent past and tune the last fully-connected layer or the last few layers, in the case of a neural network. In this direction, state-of-the-art studies have developed several drift-based approaches to determining when to update a deployed ML model. For example, Chakraborty and Adhikari (2021) investigated ML model updating in the digital twin of a multi-time scale dynamical system, where physical data was periodically used to update the ML model in learning the time-evolution of system parameters. Davis et al. (2017) studied the deterioration of ML model performance in predictive analytics for the development of acute kidney injury with ten years of patient data and highlighted the importance of routine calibration of ML models when they were incorporated into clinical decision support systems.

Q-learning is a reinforcement learning algorithm that identifies the best action given the current state of a physical system. Sun et al. (2020a) took advantage of Q-learning to sensitively quantify the deviation between the measurable response (e.g., CPU frequency of industrial devices and computer clusters or energy consumption) of a physical device (e.g., a sensor or a monitor) and its mapped value in the digital twin of an IIoT system and updated the ML model in the IIoT system to achieve a better trade-off between local update and global parameter aggregation. Wu et al. (2019a) developed error- and event-triggered mechanisms to update a recurrent neural network model using the most recent process data for model predictive control (MPC) in a chemical process subject to time-varying disturbances.

It is worth noting that ML model updating can be formulated as a parameter estimation problem, expressed as a discrete-time state-space model. In this state-space model, the ML model parameters to be updated (often a small subset of all ML model parameters) become $\boldsymbol{\theta}$ that changes by a vector of process noises (state transition equation), and the ML model becomes the measurement function that maps the ML model input and parameters to the measurement $\mathbf{y}$ (measurement equation). As new measurements from the physical system become available, Bayesian filters (Cox, 1964; Gustafsson, 2010), such as the extended Kalman filter and particle filters, can be used to solve this state-state model for online updating the ML model parameters. See a more detailed discussion in Sec. 4.2.4.

Accurate ML model prediction plays a cornerstone role in digital twins. ML model updating is just one of the mainstream ways to ensure satisfactory generalization performance of ML models in the open world. In addition to regular ML model updating, if the physics characterizing the underlying physical processes are known, incorporating these physics in the ML model can also be pursued as another viable path to improve ML models' generalizability. See a more detailed discussion of physics-informed ML in Sec. 3.4.

4.4 Fault diagnostics and failure prognostics

4.4.1 Overview of PHM

Over the past two decades, sensor data collection, fault diagnostics, failure prognostics, and maintenance decision making have drawn significant attention, collectively forming a research field called prognostics and health management (PHM). This research field has been largely promoted by (1) various workshop committees on structural health monitoring (SHM), a research field largely parallel to and synergistic with PHM, (2) the PHM Society, and (3) the rise of journals like *Structural Health Monitoring: An International Journal*. While fault diagnostics and failure prognostics form an essential, integral part of PHM, the system health management goes beyond the predictions of the RUL and provides decision support for optimal maintenance and logistics decisions by integrating also the considerations on the available maintenance resources, the operating requirements, and the economic, environmental and operational impact of different faults. In general, health management can be considered as the process of taking optimal maintenance decisions based on the outputs from diagnostics and prognostics models, while taking the relevant operational, resource and monetary constraints into consideration. In short, the goal of PHM can be considered as optimally managing the health of components and systems by minimizing the operational and economic impact of failures, and proactively controlling the direct and indirect maintenance costs Lee et al. (2014).

The concept of PHM has been exploited in different application domains, leading to a diverse set of terms, algorithms, and tools. For example, machinery health monitoring is an engineering discipline dedicated to monitoring the health of machine components, including bearings, gears, shafts, motors, and pumps, and tracking and predicting their health evolution over time. Vibration-based fault diagnostics is a mature subdiscipline of machinery health monitoring that has a much longer history than PHM, with many algorithms and tools available for processing vibration signals (Antoni and Randall, 2006) and diagnosing bearing faults (Randall and Antoni, 2011) and many applications in industrial sectors (Randall, 2021). Another example is that research on damage diagnostics and prognostics of civil infrastructures has, over time, formed an engineering discipline called SHM. As described earlier, SHM and PHM are two largely parallel yet synergistic research fields, with SHM having a much longer history than PHM. SHM primarily focuses on monitoring and detecting structural damage that alters the material and geometric properties of a structure (Farrar and Worden, 2007). In SHM, "damage assessment" is a more appropriate and widely used term than fault diagnostics. It deals with detecting, localizing, and identifying the types of, and quantifying the severity of, damage as just defined. Given that structural failure is rare and difficult to introduce, prognostics plays less of an important role than damage assessment. A third example is when PHM is applied to lithium-ion batteries, creating an active discipline of battery health management. In this discipline, a better alternative to the term "fault diagnostics" is "degradation diagnostics", which identifies and quantifies degradation modes and mechanisms driving cell-level degradation (Birkl et al., 2017). Metrics used to measure cell-level degradation are called the state of health parameters, as discussed in 4.2 (a). Two representative metrics in this regard are capacity and internal resistance, which together determine the energy and power capability of a battery cell.

A few representative review papers on failure prognostics and SHM are (1) Heng et al. (2009); Lee et al. (2014); Lei et al. (2018) for machinery health monitoring, with the first focusing on comparing traditional reliability engineering and prognostics, the second providing a balanced review on methodologies and industrial applications, and the third focusing on datasets and methodologies for health assessment and RUL prediction, and (2) Rezvanizaniani et al. (2014); Waag et al. (2014); Hu et al. (2020) for battery health management, with the first focusing on SOC and state of health monitoring as well as lifetime prognostics, the second on SOC and state of health monitoring, and the third on battery lifetime prognostics.

4.4.2 Diagnostics vs. prognostics

Before discussing diagnostics and prognostics in detail, let us first look at the key differences between fault diagnostics and failure prognostics. Fault diagnostics is focused on identifying the presence (fault detection), location (fault localization), type (fault identification) and possibly severity (fault severity assessment) of damage/defects representative of the system's *current* health state. In contrast, failure prognostics is tasked with tracking the time evolution of fault properties and projecting their progression into the *future* to yield an updated estimate of the physical system's health state or RUL, defined as how long the physical system can operate before it reaches a failure threshold or completely breaks down. Time evolution of damage or degradation is an essential factor that plays an almost dominant role in RUL prediction. In short, diagnostics gives us the current damage state, and prognostics projects that into the future to explore questions about criticality that could suggest actions such as maintenance, repair, etc.

4.4.3 Fault diagnostics approaches

Fault diagnostics is a broad topic that has been extensively studied in many application domains over the past decades. We intend to provide a brief survey of this topic in this subsection, focusing on ML-based fault diagnostics. Interested readers can gain more knowledge and insights by looking at review papers on fault diagnostics in respective domains, such as Samuel and Pines (2005); Jardine et al. (2006); Li et al. (2016); Zhao et al. (2019) for machinery fault diagnostics, Lynch and Loh (2006); Fan and Qiao (2011) for sensors in SHM applications, Farrar and Worden (2007); Fan and Qiao (2011) for structural damage detection and identification, and Berecibar et al. (2016); Li et al. (2019b) for battery state of health estimation. Furthermore, a recent special issue of the *Structural Health Monitoring* journal featured ML-based approaches (*Structural Health Monitoring* 20(4), 1353-2239, 2021.)

In ML-based fault diagnostics, ML models are either classification tools to classify the health state of a physical system (e.g., healthy, slightly damaged, or severely damaged of a roller bearing in a hydraulic motor (Shen et al., 2021)) or identify its fault type (e.g., unbalance, misalignment, mechanical looseness, rubbing, or oil whirl in a journal bearing rotor system (Oh et al., 2017)) or predictive tools to quantify a fault/-damage present in a physical system. Approaches have been developed using conventional ML and emerging DL methods. As indicated in Fig. 20, running a forward pass on conventional ML models typically involves two steps: feature extraction and classification/regression. Before training these ML models, their input features need to be properly defined. Low-dimensional features (typically less than 20) are first extracted from preprocessed sensor data and then used with other low-dimensional monitoring data (e.g., measurements of operating conditions) for fault classification or damage assessment. The commonly used feature extraction techniques include statistical feature extraction (Jegadeeshwaran and Sugumaran, 2015), principal component analysis (Misra et al., 2002; Pei et al., 2021), wavelet transform (Peng and Chu, 2004), and fast Fourier transform (Zhang et al., 2013). For example, Pei et al. (2021) used principal component analysis to extract important features from manufacturing process monitoring signals. Afterward, a support vector machine was used to map the extracted features to processing quality which is labeled as acceptable or abnormal according to the level of tolerance deviation.

While successes using conventional ML methods for fault diagnostics can be found in some application domains, these success stories have been limited to cases where the domain knowledge required to extract hand-crafted features from the raw data and select appropriate features is available. Furthermore, extracting and selecting input features can be tedious and time-consuming. It is an iterative, manual process known as feature engineering, where features are added and

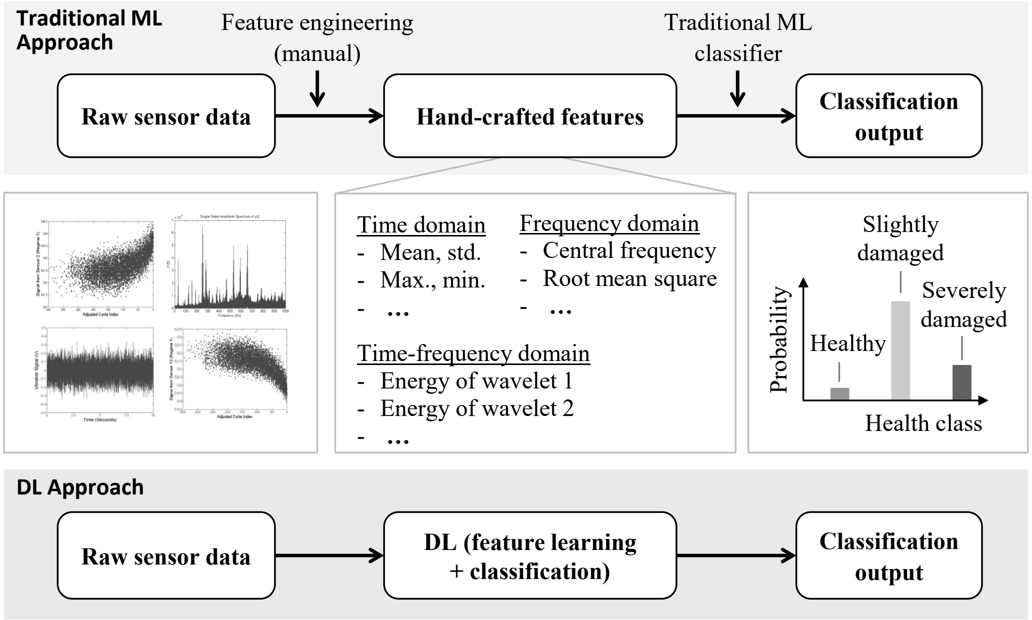


Fig. 20: A graphical comparison of traditional ML and emerging DL approaches to fault diagnostics

removed, and ML models are redefined, trained, and evaluated. Feature engineering can account for more than 90% of the total effort in building ML models.

Emerging DL methods provide end-to-end fault diagnostics capability, eliminating the need for manual feature extraction, as indicated in Fig. 20. Unlike traditional ML algorithms, DL algorithms, such as those described in Sec. 3.3.2, can automatically extract high-level, complex representations (or features) from large volumes of data. The DL models take preprocessed sensor data directly as input and produce the health class or fault/damage size as output. For instance, melt pool images and thermal images collected by cameras have been directly used as the inputs of DL models for defect detection in both additive and more conventional manufacturing processes (Liu et al., 2020a; Wang et al., 2020b; Franciosa et al., 2020).

In recent years, hundreds or even thousands of DL-based fault diagnostics approaches have been reported in the literature. Given large volumes of labeled data, DL approaches show better accuracy in fault diagnostics than their conventional ML counterparts in most applications. A review of DL methods for fault diagnostics can be found in Zhang et al. (2020b). As expected, the effectiveness of ML models for fault diagnostics is significantly affected by the quantity and quality of the data used to train the model. In particular, data from faulty states are much more difficult to collect than data from the healthy state. This data collection issue could result in data imbalance and poor accuracy in fault diagnostics. Three strategies have been investigated to address this data challenge, and they all build on known physics.

- *Synthetic data generation*: As reviewed in Sec. 3.4, a variety of physics-informed ML approaches can be used in physical system modeling to improve generalizability, to increase interpretability, and to alleviate the data-size requirement over purely data-driven ML approaches. Similarly, physics-informed ML can also improve fault diagnostic performance. If we consider that data under faulty states are very difficult to collect, " synthetic" faulty data can be generated from high-fidelity physics-based simulations and used to augment the original training dataset. The synthetic data can be generated by changing health-relevant parameters of the physics-based model (e.g., reducing material stiffness for selected regions where structural damage is intended to be injected (Kapteyn et al., 2021)). Combining the synthetic data with available real data to train and validate ML models is essentially a strategy for physics-informed ML, illustrated as Approaches

2, 3, and 5 in Fig. 10. When high-fidelity physics-based models are not available, an alternative solution to synthetic data generation is to use a DL-based generative model called a generative adversarial network (GAN). After being trained on a real dataset, a GAN can generate synthetic healthy/faulty data, which share a similar distribution as the real training dataset (Shao et al., 2019).

- *Physics-informed architecture or loss function design*: In many cases, physical domain knowledge can be leveraged to customize mathematical operations or node connections in ways that mimic essential physics and make black-box ML models more interpretable. The design of the model architecture can be designed to constrain the output of the model to better reflect the governing physics. For instance, Sadoughi and Hu (2019) proposed a physics-based (or better termed as physics-informed) CNN for fault classification of rolling element bearings. The uniqueness of their physics-informed CNN is that the kernel functions in its first convolution layer are designed based on known physical knowledge about bearings and their fault characteristics. Kim et al. (2022) developed a CNN-based classifier for gearbox fault diagnostics based on a health-adaptive time-scale representation that was built using multiscale convolutional filters. These convolutional filters were specially designed to incorporate known physics of faults, more specifically, the fault-related time and frequency characteristics in the vibration signals from a faulty gearbox. Other physics-informed ML approaches to fault diagnostics utilize frequency components of interest to customize the loss function used in ML model training, as reported in Shen et al. (2021); Russell and Wang (2022). These physics-informed ML approaches can help address the data challenge in ML-based fault diagnostics, as they can improve the generalizability of ML models to test samples unseen during model training, including those that may outside of the training distribution, as more extensively discussed in 3.4. A a result, compared to a purely data-driven ML model, a physics-informed ML model may be trained on a smaller set of faulty data to achieve a similar generalization performance.
- *Fault diagnostics based on physics-based modeling*: For physical systems in the SHM domain, such as civil infrastructure systems and large aircraft structures, or in safe-critical industrial applications, it is unlikely one would be able to collect faulty data since these physical systems are generally well-maintained, and failures are extraordinarily rare. Thus, the amount of available faulty data is very limited. Even if it is possible to collect a reasonable quantity of faulty data and build a training dataset, this dataset may not be representative of other structures operating under different loading and environmental conditions. As a result, it is challenging to build ML models readily deployable for structures or machines that these models have not seen. One way to tackle this challenge is model-based fault diagnostics taking advantage of physics-based models. This approach classifies the health state of a physical system or quantifies its fault/damage by studying model predictions under healthy and faulty conditions (Park et al., 2016), comparing model prediction with sensor data, or using sensor data to update a physics-based model (see Bayesian model updating in Sec. 4.2), without resorting to building data-driven ML models. For example, Park et al. (2016) proposed a model-based fault diagnostics method to detect faults in planetary gears. A lumped parametric model was developed to simulate transmission error signals that contained gear fault-related information. The development of this physics-based model allowed for a simulation-based investigation into how to extract fault-sensitive features from these transmission error signals, informing the implementation based on actual sensor data in practice.

4.4.4 Failure prognostics approaches

Failure prognostics approaches can be broadly divided into *physics-based*, *data-driven*, and *hybrid approaches*. An ideal example of a physics-based approach is using a physics-based model, as described in Sec. 3.2, depicting the physical processes, in combination with a degradation model, depicting the time evolution of damage or degradation parameters used in the physics-based model. Such a physics-based approach has been attempted in Ramadesigan et al. (2011), as described in Sec. 4.2.4. These ideal physics-based approaches can capture the physical processes and failure mechanisms, therefore offering good interpretability and generalisability. However, their applicability to prognostics is limited by difficulties in calibrating large numbers of parameters in the physics-based models due to, for example, poor model identifiability, as mentioned in Sec. 3.2 and 4.2 (a), an insufficient understanding of the degradation mechanisms and their long-term evolution, and high computational costs (not suitable for real-time decision making). These limitations motivated researchers to look for models that are not strictly physics-based but rather semi-empirical or empirical. Examples of semi-empirical models include the Paris' law equation describing fatigue crack growth widely used for fatigue damage prognostics (Sankararaman et al., 2011) and the square-root-of-time dependence capturing battery capacity fade due to the solid electrolyte interphase growth for battery lifetime prediction (Bloom et al., 2001; Smith et al., 2011). Empirical models do not have any physical meaning but show good agreement with data. Three representative examples of empirical models are (1) the statistical degradation models described in Sec. 3.3 (b) for general-purpose prognostics (i.e., health forecasting of a general physical system), (2) exponential degradation models, originally developed for general-purpose prognostics (Gebraeel et al., 2005) and further developed and applied to bearing prognostics (Li et al., 2015) and battery prognostics (He et al., 2011), and (3) power-law models for battery prognostics (Lui et al., 2021; Gasper et al., 2021), one of which reduces to the traditional model when the temporal power exponent equals 0.5.

Physics-based approaches, plus prognostics approaches using semi-empirical and empirical mathematical models, form a broader category called *model-based prognostics*. Figure 21 shows a diagram illustrating a model-based approach using an empirical exponential model for battery capacity forecasting and RUL prediction. This battery prognostics problem can be formulated as a state-space model for parameter estimation, as discussed in Sec. 4.2.4, and solved using particle filtering, as detailed in Sec. 4.2 (b). The model-based approach outlined in Fig. 21 is used for online estimate of the RUL of a battery cell. In a digital twin, the RUL prediction could then be used in other process and control optimization algorithms to, for example, determine the optimal time to remove the cell from its first-life application. This application of a model-based prognostics method within a digital twin framework is extensively investigated as a case study in Part 2 of this review on digital twin.

Suppose a good degradation model is unavailable, or degradation parameters are hard to estimate from observational data. In that case, data-driven approaches may be a better alternative to mode-based approaches, assuming medium to large volumes of degradation data are available. This is usually only possible in applications where life safety issues are not a concern. Data-driven approaches can be categorized further into direct mapping and time series forecasting approaches, as illustrated in Fig. 22. Direct mapping approaches bypass health forecasting and directly map features (ML model input) to RUL (ML model output) using ML models such as standard feedforward neural networks (Huang et al., 2007; Tian, 2012), deep CNN (Li et al., 2018, 2019a), and LSTM recurrent neural networks (Yuan et al., 2016; Huang et al., 2019). Some variants of direct mapping approaches do not output RUL and instead estimate a health indicator (e.g., an overall measure of engine/milling machine health (Malhotra et al., 2016) or bearing health (Guo et al., 2017), battery capacity (Shen et al., 2020)), or a battery degradation parameter such as an electrode aging parameter (Tian et al., 2021; Thelen et al., 2021)). ML-based forecasting approaches predict how a health indicator or reliability metric (such as time to failure or survival probability) evolves beyond the current time as an intermediate step for RUL prediction. ML models used for forecasting include support vector machine (Pham et al., 2012), relevance vector machine (Wang et al., 2013), Gaussian process regression (Richardson et al., 2017), feedforward neural networks (Fink et al., 2014), and LSTM recurrent neural networks (Nemani et al., 2021), just to name a few. Applications of DL to PHM are well-reviewed in Khan and Yairi (2018); Zhao et al. (2019); Fink et al. (2020), with the first and second papers focusing on machinery health monitoring, the second paper providing a comparative study on the tool wear prediction performance of ten ML and DL algorithms, and the third paper focusing on successes of DL in various application domains including PHM.

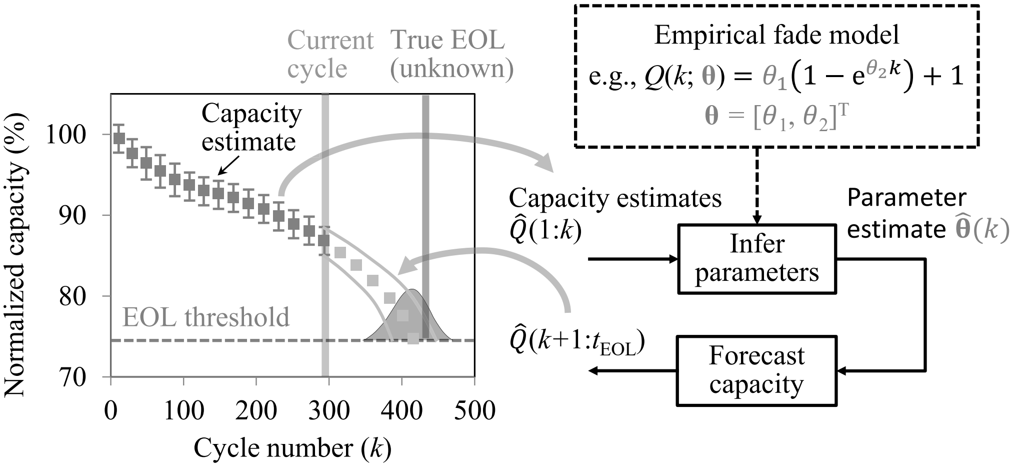


Fig. 21: A schematic illustrating model-based prognostics for battery capacity forecasting and RUL prediction. Q denotes the capacity of a battery cell normalized by its initial capacity, and k denotes the cycle number. EOL in the figure stands for end of life.

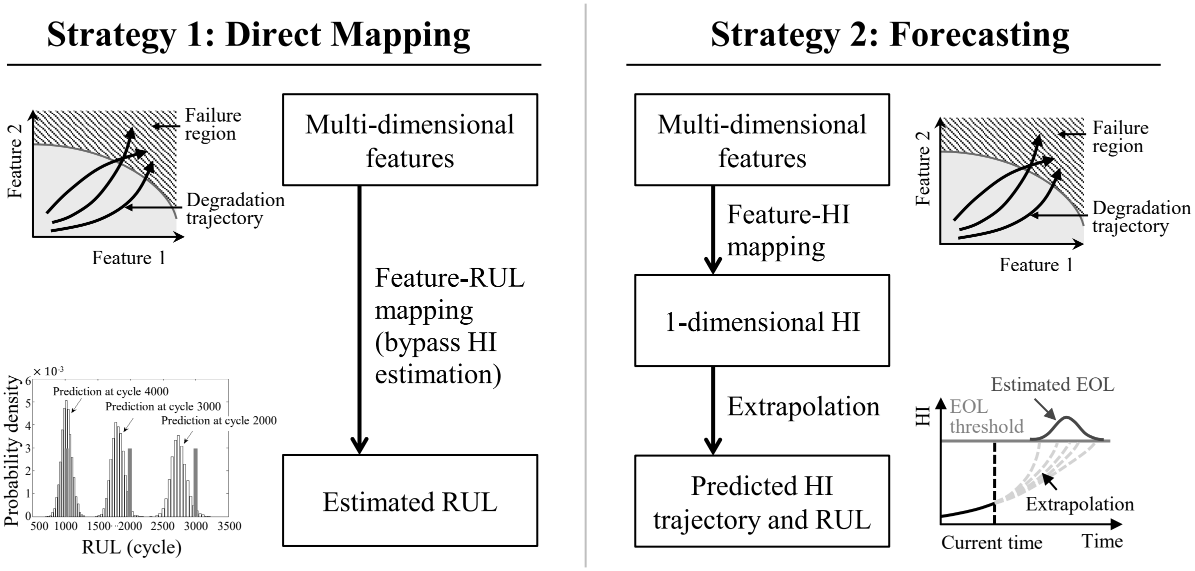


Fig. 22: A graphical comparison of two distinct strategies for data-driven prognostics. HI and EOL stand for health indicator and end of life, respectively.

Data-driven ML approaches can discover complex degradation patterns relevant to RUL prediction from large volumes of data, are simpler to implement than physics-based approaches, and are agnostic to the physical and degradation processes. However, large ML models can have many more parameters than physics-based models and require large training datasets to ensure low risks of overfitting. In reality, collecting large and representative run-to-failure datasets can be costly and time-consuming. Another well-known issue with ML models is that they have difficulties in generalizing to out-of-distribution data unseen during model training, as also discussed in Sec. 3.4 and 4.3, and may produce predictions that do not make physical sense. In conclusion, the performance of purely ML models is limited by the quality and quantity of the available training data. Without consideration of physics, the accuracy numbers on a test dataset that falls within the training data distribution may not hold for other datasets outside the training distribution.

As discussed above, purely physics-based and data-driven approaches have complementary advantages that can be leveraged to tackle the limitations. For example, physics-based models offer opportunities to generate synthetic data that can help alleviate the data-size requirement of data-driven approaches, extending the applicability of these approaches to cases where run-to-failure trajectories are scarce. Using physics-based models to generate synthetic training data is illustrated as Approaches 2, 3, and 5 in Fig. 10. Another example is that data-driven ML models can learn complex bias functions from data to compensate for discrepancies in physics-based models. This strategy in combining physics-based and model-based approaches is illustrated in Fig. 10 as Approaches 4 and 5, where the bias functions respectively capture the prediction inaccuracy of a physics-based model (Approach 4) and the prediction inaccuracy of an ML model trained with physics-based synthetic data (Approach 5). All the approaches described in Sec. 3.4 can be adapted for failure prognostics, with some successes recently reported in Chao et al. (2022); Kohtz et al. (2022); Yucesan and Viana (2022).

4.5 Ontology-based reasoning

Ontology maps and knowledge graphs provide a straightforward way to represent empirical knowledge in a structured manner. The unified knowledge engineering and representation through descriptive logic lays a firm foundation for knowledge reasoning over knowledge graph in the semantic space. In practice, they are primarily implemented through ontology markup languages (Chen, 2010), such as Web Ontology Language, XML Schema, Resource Description Framework Schema (RDFS).

The proper construction of an ontology map and knowledge graph facilitates knowledge inference along the semantic dimension (Chen et al., 2020a). In general, knowledge reasoning methods can be categorized into three classes.

1. **Logic rules-based reasoning**. Rules-based reasoning manifests in several forms: (a) first-order predicate logic rules, such as First-Order Inductive Learner (FOIL) (Schoenmackers et al., 2010), that is to use the predicate as the basic unit for knowledge reasoning; (b) manually inject user-defined or simply learned rules in knowledge graph to generate new knowledge, such as the inference component in the Never-Ending Language Learning system (NELLs) (Mitchell et al., 2018). (c) to incorporate frequently present patterns, constraints or paths to extract new knowledge. That is to reason with the Web Ontology Language. (d) identify path rules and inject them into knowledge graph to facilitate reasoning. For example, Lao and Cohen (2010) utilized a path ranking algorithm to identify the associations between edge types and the instances of edge types, and exploited such associations to predict missing edges in the graph.
2. **Distributed representation-based reasoning**. This line of studies aim to project the basic elements in knowledge graph (i.e., entities, relations, attributes) into a continuous vector space through embedding-based approaches (Nickel et al., 2011), including tensor decomposition, distance-based mapping, and semantic matching models. The distributed representation allows to discover latent rules and connections in the transformed space.
3. **Neural network-based knowledge reasoning**. In recent years, more and more studies have explored the representation learning ability in deep neural networks for knowledge reasoning. Specifically, the CNN-based learner enables to account for the fine-grained multi-source heterogeneous information encoded in entity description (Xie et al., 2016); the recurrent neural network is capable of reasoning about multi-hop relations in knowledge graph (Neelakantan et al., 2015). These unique advantages have positioned neural network as an appealing paradigm for knowledge inference in the semantic space.

Over the past few years, there has been an increasing investigation of ontology techniques for data management as well as knowledge representation and inference in digital twins. For example, Ladj et al. (2021) combined business rules with the knowledge deriving from industrial data analytics to build ontology-based knowledge model for incident detection in the machining process, and the knowledge base served as an inference platform to detect detrimental incidents. Lim et al. (2020b) employed knowledge graph-based method to accommodate heterogeneous data sources, such as asset ontology, environment ontology, and system ontology. The comprehensive data integration in knowledge graph drives the inference of smart solutions for asset configuration, resource planing and component maintenance. Liu et al. (2021d) developed multiscale product quality knowledge model to characterize the relationship between product quality and the product quality factors explicitly.

5 Virtual-to-physical (V2P) twinning enabling technologies

Figure 23 summarizes the modeling and twinning methods by application in different phases of a physical system's life cycle. As illustrated in this figure, there are numerous ways of establishing V2P connection, such as system reconfiguration, process control, production planning, maintenance scheduling, and path planning. In this section, we mainly concentrate on two classical V2P twinning enabling technologies, namely model predictive control and predictive maintenance.

5.1 Model predictive control

Model predictive control (MPC) includes a wide range of advanced control methods, which in common use a model to predict the future behavior of the process and determine the optimal control within a set of constraints. MPC is an essential component for smart manufacturing, which enables not only the optimal use of resources for high-quality products, and but also quick production responses to changes in market demands and supply chains (Lu et al., 2016). It is one of the most widely used methods to establish the V2P connection in a digital twin.

Figure 24 shows a general structure of MPC, including a process model, an objective function, process measurements, constraints and sampling points (Nikolaou, 2001).

Assuming an arbitrary process with a state space representation:

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{f}(\mathbf{x}(k),\ \mathbf{u}(k)), \\ \mathbf{y}(k+1) &= \mathbf{h}(\mathbf{x}(k+1)). \end{aligned} \tag{10}$$

MPC minimizes a user-defined cost function J,

$$\min_{\mathbf{u}} \sum_{t=0}^{k} J(\mathbf{x}(t),\ \mathbf{u}(t)), \tag{11}$$

in which $J(\mathbf{x}(t),\ \mathbf{u}(t)) : \mathbb{R}^s \times \mathbb{R}^c \to \mathbb{R}$ is the cost function at t.

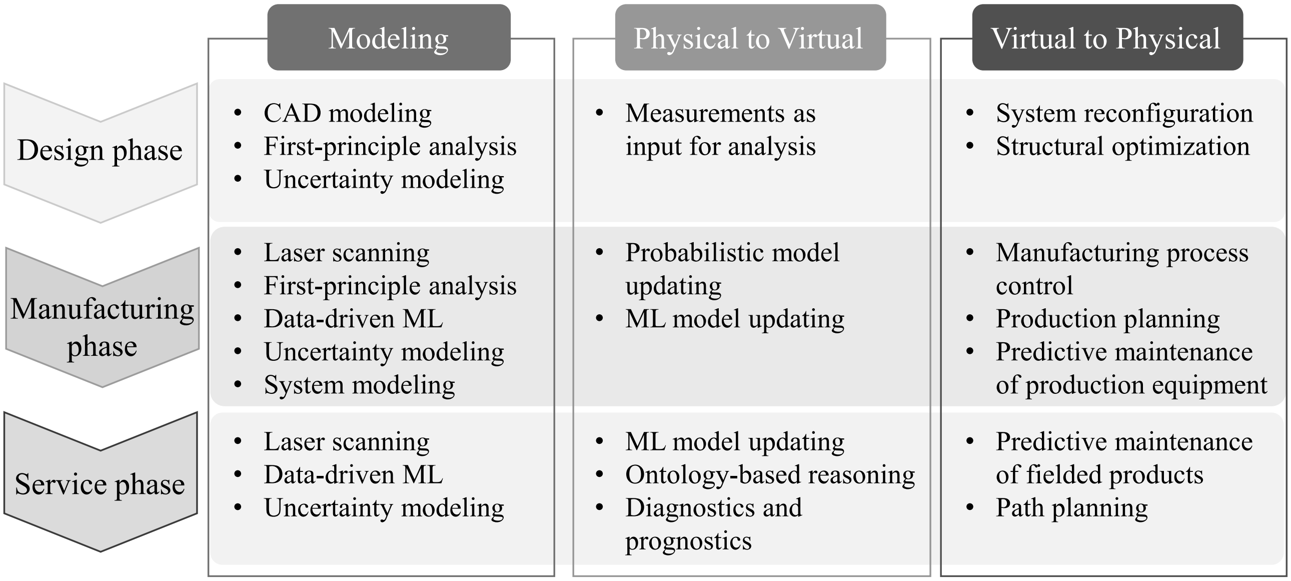
For a trajectory following process control problem, the objective functions can be any norm of the tracking error between the reference vector $\mathbf{r}$ and the model output $\mathbf{y}$, as shown in Eq. (12). The optimization problem is defined under hard state and input constraints:

$$\begin{aligned} \min_{u} \quad & \sum_{i=N_1}^{N_2} \|\mathbf{r}(k+i \mid k) - \mathbf{y}(k+i \mid k)\| \\ s.t. \quad & \mathbf{u}_{lb} \le \mathbf{u}(k+j \mid k) \le \mathbf{u}_{ub} \\ & \mathbf{y}_{lb} \le \mathbf{y}(k+j \mid k) \le \mathbf{y}_{ub} \\ & \forall\ i \in \{N_1, ..., N_2\}\ \ and\ \ j \in \{0, ..., N_u\}, \end{aligned} \tag{12}$$

where $N_1 \le N_u \le N_2$.

The iterative, finite-horizon optimization of a process can be further illustrated in Fig. 25. At time t_k the current process measurements are acquired and a cost minimizing control strategy is computed to minimize the error between the system output y and the given reference r for a horizon N_2. N_2 must be long enough to represent the effect of a change in the control commands u on the control variable y. If there a pure delay exists, the sum of the objective function starts from N_1. Only the first value of the optimal control commands is implemented, then the process measurements are sampled again and the optimization is repeated, yielding new sets of control commands and new predicted state path. The prediction horizon keeps being shifted forward so MPC is also called receding horizon control.

MPC has become the most widely implemented advanced process control technology and is offered by major automation suppliers, with applications in process industries such as refining, petrochemicals, paper and food processing (Industry; Automation) . In the last decade, MPC is extended and also adopted in power plant control, build thermal management and discrete manufacturing (Deng et al., 2014; Farzan et al., 2014;



	Modeling	Physical to Virtual	Virtual to Physical
Design phase	• CAD modeling • First-principle analysis • Uncertainty modeling	• Measurements as input for analysis	• System reconfiguration • Structural optimization
Manufacturing phase	• Laser scanning • First-principle analysis • Data-driven ML • Uncertainty modeling • System modeling	• Probabilistic model updating • ML model updating	• Manufacturing process control • Production planning • Predictive maintenance of production equipment
Service phase	• Laser scanning • Data-driven ML • Uncertainty modeling	• ML model updating • Ontology-based reasoning • Diagnostics and prognostics	• Predictive maintenance of fielded products • Path planning

Fig. 23: Classification of modeling and twinning techniques according to phase of a physical system's life cycle

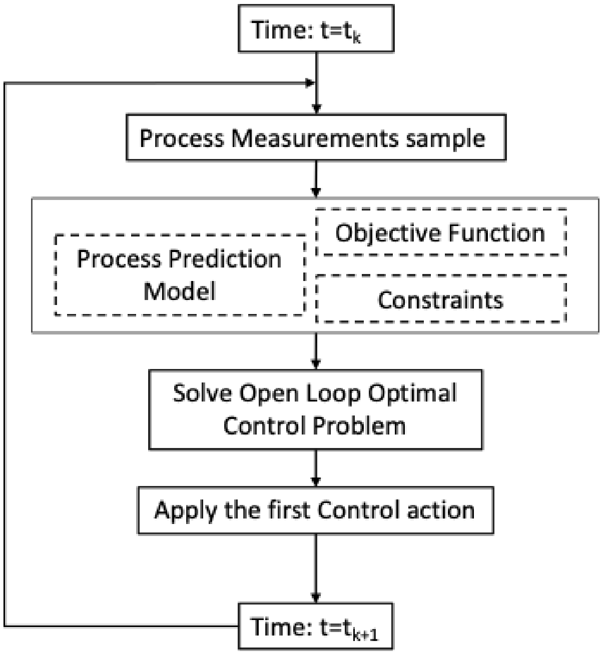

Time: t=t$_k$
Process Measurements sample
Process Prediction Model
Objective Function
Constraints
Solve Open Loop Optimal Control Problem
Apply the first Control action
Time: t=t$_{k+1}$

Fig. 24: MPC structure Nikolaou (2001)

Swief et al., 2019). In the era of smart manufacturing, the process and manufacturing industries are embracing tremendous opportunities brought by the Internet-of-Things (IoT) and advanced data analytical technologies, which also enhance the application of MPC by expanding the objective function to more some general performance measure (Ellis, 2015), and by combining machine learning techniques in process modeling (Essien and Giannetti, 2021). Economic MPC employs cost function connected to the economics of the considered process which makes economic MPC well suited as a tool to achieve the goals of smart manufacturing. Machine learning-based MPC is

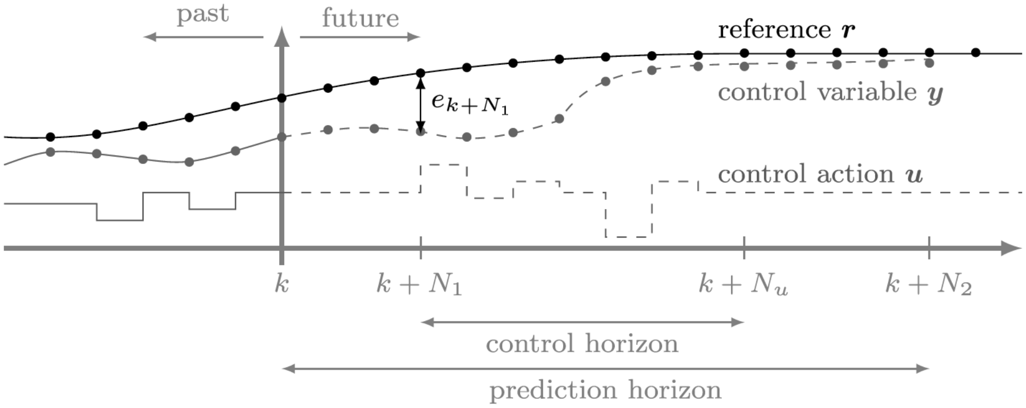

past
future
reference $\boldsymbol{r}$
e_{k+N_1}
control variable $\boldsymbol{y}$
control action $\boldsymbol{u}$
k
$k+N_1$
$k+N_u$
$k+N_2$
control horizon
prediction horizon

Fig. 25: Iterative Horizon Receding Control of MPC Schwenzer et al. (2021)

reported to use recurrent neural network modeling approaches for a general class of nonlinear dynamic process systems to improve prediction accuracy and control performance (Wong et al., 2018). There are also efforts reported to integrate a DL architecture with MPC for robot manipulation control (Lenz et al., 2015). Rather than developing a dynamics model from first principles, the future MPC will more likely to learn digital models using a novel deep architecture and learning algorithms directly from data. For example, some of the data-driven models summarized in Sec. 3.3 can be used as predictive models for MPC in digital twins.

5.2 Predictive maintenance

A recent industry-wide shift called *Industry 4.0* capitalizes on increased interconnectivity and automation to make machines on a production floor smarter. Benefits of this digital transformation to Industry 4.0 include increased productivity, improved workflow efficiency, and improved product quality. A key enabling step of Industry 4.0 is predictive maintenance, mostly discussed in the manufacturing domain. Predictive maintenance is a proactive approach to equipment maintenance that centers on (1) identifying signatures in sensor data (continuous monitoring or periodic inspections) indicative of changes in equipment health (see fault diagnostics discussed in Sec. 4.4.3), (2) predicting when a machine, component or part might fail (see failure prognostics discussed in Sec. 4.4.4), and (3) scheduling maintenance work during planned downtime just before equipment failure, sometimes referred to as "just-in-time maintenance" (Lee et al., 2013a,b).

Now let us answer a simple question: why is predictive maintenance needed, for example, for a rotating machine (e.g., motor, pump, and fan)? The answer is quite simple. The unexpected failure of this rotating machine often incurs high maintenance and downtime costs, reduces customer satisfaction with a produced good, and may even cause human injuries and fatalities. These consequences also impact the machinery manufacturer by tarnishing their reputation and potentially putting them at a competitive disadvantage. Therefore, it will be value-added to develop, implement, and deploy predictive maintenance solutions, especially justifiable for high-value equipment whose failure cost is high. Note that predictive maintenance differs from the more traditional *reactive maintenance*, which performs maintenance work after an equipment failure and may lead to costly unexpected downtime, and *preventive maintenance*, which performs maintenance work at a fixed time interval, typically more often than necessary, and may cause a waster of time and resources due to unnecessary maintenance.

A typical ML pipeline for building and deploying a predictive maintenance solution for a rotating machine is shown in Fig. 26 and consists of four main steps.

1. **Acquire sensor data (P2V):** This first step identifies critical machine assets for condition monitoring and installs sensors on these assets to measure signals that are sensitive to machine health. For example, sensor signals sensitive to

bearing health include acoustics, vibration, oil wear particle count, and temperature, ranked from the earliest to latest in detecting a bearing fault. Note that in SHM, the placement of a sensor network needs to be optimized to maximize the amount of useful information for structural damage assessment, as discussed in detail in Sec. 2.2.1 of Part 2 of the review paper.

2. **Preprocess data (P2V):** Field data is rarely clean; it contains outliers and noise that need to be removed to avoid corrupting ML model performance. An example in bearing health monitoring is that vibration signal processing is almost always needed to denoise vibration signals by, for example, designing frequency-domain filters used to remove background noise irrelevant to machine faults (Antoni and Randall, 2006). Another example is that outliers in battery voltage and current measurements must be removed before they are fed into ML models for capacity estimation or RUL prediction enabling battery health management.
3. **Train ML models (P2V):** This step trains ML models to classify the machine condition as healthy/faulty (green/red) or healthy/lightly damaged/heavily damaged (green/yellow/red) and estimate the RUL of the machine. For both fault classification and RUL prediction, the predictive uncertainty of the ML models needs to be estimated, accurately reflecting the models' confidence in making a health class/RUL prediction (see Sec. 2.1.1 in Part 2 of the review paper for a detailed discussion on ML model uncertainty). A probabilistic estimate of RUL can be in the form of a mean RUL and an associated confidence interval representing the predictive uncertainty.
4. **Optimize maintenance decision making (V2P):** This last step optimizes maintenance schedules based on the predictive information from Step 3. In practice, what is preferred is to repair or replace equipment during a scheduled downtime just before failure occurs, without affecting the chance of meeting production goals. Maintenance decisions need to be made from a risk perspective, given that ML model predictions are associated with varying degrees of uncertainty. A dedicated review of maintenance schedule optimization will be given in Sec. 2.2.3 (c) in Part 2 of the review paper.

Digital twin can be an enabler of predictive maintenance. As discussed earlier in Sec. 4.4.3 and 4.4.4, one of the main issues with diagnostics/prognostics is the lack of faulty/run-to-failure data (i.e., data acquired from a physical system whose health state or RUL is known). This data challenge can be addressed by using a physics-based model in a digital twin, calibrated offline and updated online (Sec. 4.2), to generate synthetic faulty/run-to-failure data. This strategy for tackling the data challenge has already been discussed in Sec. 4.4.3.

An example where a digital twin can help with the online phase of predictive maintenance is that a physics-based model in a digital twin, considering both physical and degradation processes, takes sensor data as input, estimates digital state variables that significantly affect lifetime, and predicts RUL. Sometimes, ML models can assist physics-based models in estimating the digital state variables. This strategy is similar to the ML-assisted approach (Approach 6) shown in Fig. 10, except that ML models may not always be used in the digital twin. A lot of success stories have been reported in the manufacturing industry on this strategy, for example, Ansys Twin Builder and PTC thingworx collaboratively working on estimating the maximum temperature of the rotor, stator, and case of an electric motor based on sensor measurements to predict the motor's RUL (ANSYS).

Another industrial success story is the predictive maintenance alarm system in MATLAB developed by Baker Hughes for the positive displacement pumps on their trucks. The core of this system is the pump health monitoring software that analyzes real-time pressure, vibration, and timing signals and uses a trained neural network to predict pump failures. It is shown that the software can classify pump health into one of three health classes, namely "normal operation", "monitor closely", and "maintenance needed", enabling better maintenance planning. However, it is unclear whether the software has predictive capabilities (i.e., whether it can predict the RUL of a pump). As the emergence of IIoT platforms facilitates the industry-scale adoption of digital twins, we can expect the number of industrial success stories of predictive maintenance to continue increasing rapidly. Figure 27 illustrates an industry-grade IIoT platform for proactively

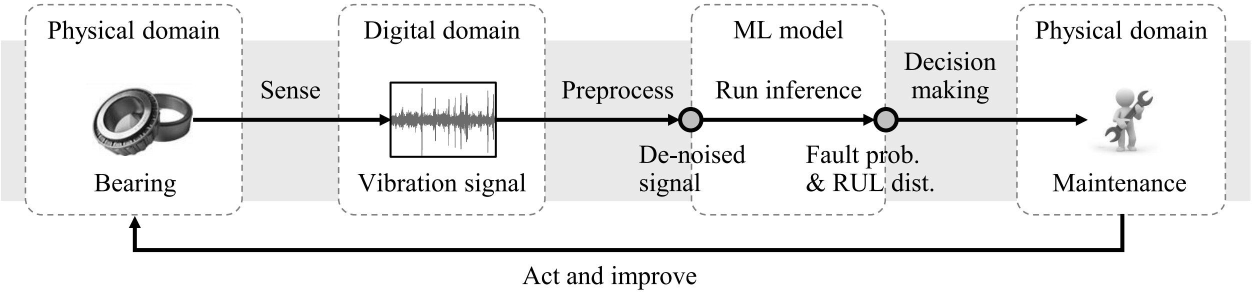


Fig. 26: An overview of an ML pipeline for predictive maintenance

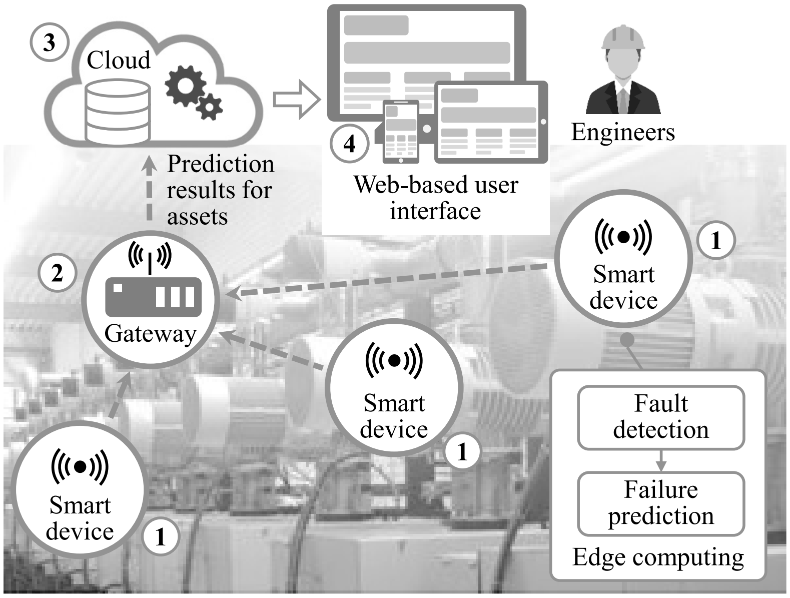


Fig. 27: An IIoT platform for predictive maintenance of industrial assets

predicting and preventing the failure of rotating machinery on a manufacturing floor. In this IIoT platform, a network of smart devices perform vibration analysis, fault detection, and failure prediction on the edge, and then send detection and prediction results to a web-based user interface that provides real-time analytics, dashboards, and alert capabilities to the maintenance and reliability engineers responsible for keeping rotating equipment in service. Maybe talk about the possibility of building centralized digital twins and distributed digital twins on the edge and sharing ML model information among clients without sharing data using federated learning in Sec. 6.1.

5.3 Real-time perception and decision making

After sensor measurements characterizing the state of a physical system are transmitted to the digital counterpart, simulation and optimization models enable the **P2V and V2P** connections within the digital twin. Specifically, simulation enables precisely modeling of the behavior of the physical system under uncertainty (P2V), and optimization drives control, maintenance, and decision making on the physical asset (V2P) given its current state. Ideally, the simulation and optimization algorithms in the digital model will be performed in real-time and will not have significant latency. However, this is not the case, as simulation and optimization are known to be time-consuming and computationally expensive, which substantially hampers their ability to provide timely results in digital twins which operate on physical systems with short time scales. To this end, the power of ML can be fully exploited to alleviate many of the inherent issues incurred by using classical simulation and optimization models. Various ML techniques can be used to facilitate real-time or near real-time (sometimes termed just-in-time) perception and decision making. Specifically, ML can be leveraged to infer the parameters of or replace the simulation model for accurately modeling the physical system (Goodwin et al., 2022). For example, Zohdi (2020) used ML to ascertain the parameters of a multi-submodel system for fast and accurate simulation of fire propagation in order to support continuous update of the digital twin of the physical counterpart in near real-time. In each case, the simulations are expedited by using pre-trained ML models which are very computationally inexpensive to evaluate. These examples are a few of many where ML can be used to speed up simulation operations in digital twins.

Regarding optimization, the expensive iterations in classical optimization algorithms can be replaced by an end-to-end ML-based optimization model proxy. In this method, the ML-based optimizer typically adopts a predict-then-repair paradigm. The basic idea is to first reformulate the optimization problem as a learning problem and then incorporate fast feasibility-restoration procedures to guarantee the feasibility of the produced solution (Bengio et al., 2021; Barry et al., 2022). ML-based optimization proxies have been shown to surpass conventional optimization algorithms by a large margin in computational speed. For example, Van Hentenryck (2021) showcased the role of ML in approximating the optimal solutions of power flow problems and the great potential of ML-based optimization proxy in speeding up existing algorithms. Chen et al. (2022) proposed an end-to-end DL model that can predict an optimal solution for Security-Constrained Economic Dispatch (SCED) in milliseconds to meet the need of power gird in real-time operations. Creating this end-to-end DL model overcomes the fundamental limitation of classical optimization techniques in computational efficiency.

Ultimately, machine learning will play a large role in enabling real-time simulation, optimization, and control of physical systems. Researchers are actively investigating methods to increase the speed of physics-based simulations and traditional optimization algorithms. These techniques will pave the way for digital twins which operate on the millisecond and sub-millisecond timescales.

6 Perspectives on modeling and twinning in digital twins

6.1 Federated learning in digital twins

For data-driven digital twins, a single system or component may not have sufficient condition monitoring data to train a representative model. This is particularly due to the fact that industrial and infrastructure systems are operated under varying conditions. Thus, monitoring a system for a limited period of time may not cover all possible expected operating conditions. Moreover, since faults or critical system states are rare, one single system may not observe a sufficient number and type of faults to train the model. Therefore, data from similar systems can be leveraged to improve the performance and generalization of the developed model.

Accumulating data from multiple similar systems can eventually result in large-scale datasets that enable representative data-driven digital twins to perform well under all relevant operating conditions. Generally, compiling such representative datasets could be achieved through sharing the data between different stakeholders, for example between different operators of similar systems. However, even without direct competition, companies are often reluctant to share data mainly due to the concern of disclosing proprietary company information. Federated learning could potentially overcome that concern enabling to take advantage of the benefits of a shared model without the need of data disclosure. Federated learning is a privacy preserving learning concept. It enables to benefit from the local data subsets (e.g., of several power plant operators) without the need of sharing the data (McMahan et al., 2017; Li et al., 2020a). It is a decentralized machine learning technique that enables to train a neural network across multiple local datasets without exchanging or sharing data samples across the decentralized nodes. A global model is constructed by aggregating (typically averaging) the parameters (either gradients or weights) of the locally trained models. Hence, it can make predictions based on the experience of the entire fleet but without the stakeholders having access to each other's data. Each individual system retains its own collected condition monitoring data on the edge, without sharing it with other systems. It gets a current version of the model, improves it based on its collected data and sends the updates to the model. Two different concepts of federated learning are typically distinguished: cross-device federated learning (also referred to as sample-based or horizontal federated learning) (Yang et al., 2019) and cross-silo federated learning (also referred to as feature-based or vertical federated learning) (Yang et al., 2019). In the context of digital twins, in fact, particularly the horizontal federated learning concept is the most relevant. However, there are also contexts where vertical federated learning could be applied.

Privacy preservation is an essential characteristic of federated learning. However, even though data is not directly shared, only sharing model updates during the training process can still potentially reveal sensitive information (Zhu et al., 2019a). Different privacy preserving approaches have been proposed that provide privacy guarantees and prevent potential indirect data leakage (Truex et al., 2019; Chen et al., 2020b; Yin et al., 2021). Federated learning has not yet been broadly applied for digital twins. Federated learning will then face similar challenges as other data-driven approaches, such as the lack of representative datasets, differences in operating conditions and system configurations between different entities of the digital twins. One of the potential directions to tackle these challenges is combining federated learning and transfer learning.

6.2 Domain adaptation in digital twins

Data-driven digital twins can show a significant performance drop if they are applied under operating or environmental conditions that they have not been trained on. However, evolving operating conditions are very common in practical applications. Moreover, if a data-driven digital twin is applied across different units of a fleet, the performance may also drop significantly due to discrepancies either in system configurations or operating regimes of different units of a fleet. It is, however, unrealistic to assume that sufficient data can be collected for all units and for all operating conditions. Different operating and environmental conditions and different system configurations result in the so-called domain shift or distributional shift, a change in the data distribution between the source (training) and the target (testing) dataset (Fink et al., 2020; Li et al., 2022), which is also described in Sec. 4.3 in the context of machine learning. Typically, no labels are available in the target dataset, so that fine-tuning of the trained model cannot be performed. Further to the two scenarios of domain shifts caused by differences in operating conditions or by differences between different units of a fleet, a third scenario, particularly relevant to digital twins is the synthetic to real gap existing between synthetically generated data stemming potentially from model-based simulators and the actual observation (Bousmalis et al., 2018; Wang et al., 2021a; Li and Lee, 2021).

Domain adaptation, a subfield of transfer learning, aims at overcoming the differences between the domains so that the algorithms are able to generalize across domains and learn domain-invariant features. Domain adaptation solves the domain shift problem by finding a mapping from the source data distribution to the target distribution. Different approaches for domain adaptation have been proposed. One of the most obvious approaches to overcome the domain gap is distribution alignment approaches by statistics matching, including Adaptive Batch Normalization (AdaBN) and Automatic DomaIn Alignment Layers (AutoDIAL), using the statistics of source and target mini-batches at different layers to align the two distributions (Long et al., 2015; Yang et al., 2016; Carlucci et al., 2017). Some of the other commonly applied approaches have been relying on the principle of minimizing a divergence-based criterion between source and target distributions, including such criteria as Maximum Mean Discrepancy (MMD) or Wasserstein Discrepancy (Yan et al., 2017; Lee et al., 2019). Due to the superior performance on many different tasks, adversarial domain adaptation has recently emerged as one of the most frequently applied approaches to confront domain shift (Ganin and Lempitsky, 2015; Tzeng et al., 2017; Ganin et al., 2016). A domain discriminator is then applied to ensure that the extracted features become indistinguishable between the two domains, while the performance on the main classification task is maximized.

Domain adaptation has been recently broadly applied to many different tasks in prognostics and health management, partly also for digital twins (Li et al., 2022; Liu et al., 2020b). One of the additional challenges that is typically encountered in real applications is that the label space of the source (training) is often not identical with that of the target (testing) dataset. This is particularly relevant in the context of fault diagnostics, where different systems may have experienced different fault types within the (short) observation time period. Possible setups in real applications are that some classes are missing in the target dataset (Partial Domain Adaptation), some classes may also be missing the source datasets (OpenSet Domain Adaptation). Or we may be even facing only a partial overlap of the label space between source and target datasets, resulting in a combination of the two types (Partial & OpenSet Domain Adaption) (Boris et al., 2021). Some approaches have been recently proposed to tackle such challenges (Li et al., 2020c; Wang et al., 2020c; Li and Zhang, 2020; Rombach et al., 2022), on the one hand by overcoming misalignment in case of different label spaces and on the other hand by developing generative models for a controlled generation of the missing fault types. While domain adaptation has been flourishing for classification tasks, its application to regression has been rather limited (Boris et al., 2021). This has been also reflected in the tasks tackled in the field of prognostics and health management, where the main application field has been fault diagnostics, with only few research works addressing prognostics tasks as well (Wu et al., 2019b; da Costa et al., 2020). Similar to classification tasks, domain adaptation for regression is also facing the challenge of label discrepancy between the source and the target label spaces (Boris et al., 2021).

6.3 Deep reinforcement learning in digital twins

Reinforcement learning is concerned with learning an optimal policy for sequential decision making problems through an agent's interactions with the environment over time by trial and error (Mnih et al., 2015). Towards policy learning, reinforcement learning not only considers the immediate reward of an action, but also takes into account the accumulative reward resulting from the action in a discounted manner. In general, reinforcement learning is composed of several key components: state space, action space, policy to map from state space to action space, state transition probability, and reward function. The advantage of reinforcement learning in long-term cumulative reward modeling has put it in a good position to tackle complex sequential decision making problems.

Deep neural networks have recently been used to approximate such functions as value functions (value function is used to predict the expected cumulative reward indicting how good a state-action pair is), policies (policy describes which action to take given a specific state), reward functions, and state transition function. For example, the well-known deep Q-network (DQN) developed by Mnih et al. (2015) approximates the state-value function in a Q-Learning framework with a neural network. Similarly, deep deterministic policy gradient (Silver et al., 2014) concurrently learned the action-value function and the policy.

For digital twins, deep reinforcement learning has only recently started to be applied. There are two main directions that have been pursued: 1) Establishing the P2V connection for model updating; 2) Building upon the operational digital twin, deep reinforcement learning has been used for control tasks and decision support, in particular due to its learning capabilities. As one example for the P2V connection, Tian et al. (2022a) trained an agent with Lyapunov-based maximum entropy deep reinforcement learning to perform model calibration and infer the uncertain model parameters of the physics-based model, such that the response of physical model matched the observed data. The proposed framework provides a similar performance for the model calibration as model-based approaches, such as UKF, but provides real-time update capability. As an example for the the P2V connection, Xia et al. (2021a) investigated the application of deep reinforcement learning in industrial process control for smart manufacturing, where deep reinforcement learning was trained to learn an intelligent operations scheduler. Some other examples include a deep reinforcement learning algorithm trained to find the optimal off-loading policy in an IoT digital twin framework with a goal of reducing energy consumption while enhancing data processing efficiency Dai et al. (2020). Moreover, Khan et al. (2020) provided an interesting perspective postulating that the digital twin as a technology for decision automation could be treated as a deep reinforcement learning model, where the physical system provides states and rewards information to the digital system via the P2V connection and digital system recommends actions to physical system to maximize the long-term reward.

Future research directions could make progress both directions: on the one hand integrating deep reinforcement learning more in the learning process of data-driven (and hybrid) digital twin, such for example for meta-learning tasks; on the other hand deep reinforcement learning is essential for interweaving more the digital twin and decision support, respectively control tasks enabling to harness more benefits from digital twin technology. One of such directions could be fore example, prescriptive operations, where deep reinforcement learning is used to prescribe an optimal course of actions with a potential objective to prolong the remaining useful lifetime Tian et al. (2022b).

7 Conclusion

In this paper, we propose a five-dimensional definition for digital twin based on how data flows between the physical and virtual systems. The five dimensions are the: physical system, digital system, an updating engine (P2V), a prediction engine (V2P), and an optimization dimension (OPT). The five-dimensional digital twin provides a clear and complete picture on the different level of interactions between the physical system and its digital counterpart (i.e., data exchange, modeling, actions). Following the five-dimensional digital twin definition, we comprehensively review the underlying techniques that are commonly used in the state-of-the-art literature to enable each dimension of the digital twin and present our findings in a series of two papers. This paper covers the enabling techniques for three specific dimensions of digital twin: physical system modeling, P2V, and V2P. In the following paper, we will concentrate on articulating how to fully incorporate UQ and optimization in the three dimensions of digital twin described in this paper.

Specific to this paper, we provide a comprehensive and rigorous examination on various means of modeling different aspects of a physical system (e.g., geometry, structure, system dynamics, relationship), the key P2V enabling techniques, and the actions and benefits (e.g., model predictive control, predictive maintenance) enabled by V2P twinning techniques across different phases of a physical system's life cycle. The accurate representation of a physical system together with the constant interactions enabled by P2V and V2P techniques form a closed loop to drive the implementation of digital twin in practice. In Part 2 of the review paper, we investigate how to ensure robust performance of a digital twin model by incorporating UQ and optimization techniques into the underlying enabling technologies.

Appendix A: A generic particle filter algorithm

Particle filtering is a widely used Bayesian filtering technique for state estimation in generic state-space models and DBNs. It is a key enabler in many digital twin applications where probabilistic model updating with state estimation plays an essential role. Let us briefly look at how particle filtering works. Particle filters represent a family of algorithms that recursively execute Bayesian filtering through sequential Monte Carlo simulation (Arulampalam et al., 2002). In a particle filter, the marginal posterior of a state at the current time step k is approximated using a large set of weighted particles. This approximation takes the following form

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \approx \sum_{j=1}^{N_p} w_k^j \delta(\mathbf{x}_k - \mathbf{x}_k^j), \qquad (13)$$

where $\mathbf{x}_k^j$ is the jth particle of the state x at the kth time step, w_k^j is its associated weight, NP is the total number of particles, and δ is the Dirac delta function. Algorithm 1 depicted in Fig. 28 gives the pseudo-code of a generic particle filter (Hu et al., 2018; Cappé et al., 2007). Three key steps of the particle filtering procedure are described as follows:

- State Transition: Each particle of the state transitions forward in time by one step (from the previous, $(k–1)^{th}$ step to the current, k^{th} step according to the state transition equation in Eq. (13) (Line 5). These particles are equally weighed $(1/N_P)$, and they form the prior of $\mathbf{x}_k$ before the current measurement is used.
- Weight Evaluation and Normalization: The weight of each particle is updated using the likelihood of the current measurement given the state values of the particle, i.e., $\mathbf{w}_k^i \propto \mathbf{w}_{k-1}^i p(\mathbf{y}_k|\mathbf{x}_k^i)$ (Line 6). The likelihood is calculated by comparing the measurement distribution predicted by the measurement equation $\mathbf{g}$ in Eq. (13) to the actual measurement $\mathbf{y}_k$. After that, the updated weights are normalized, and the normalized weights sum up to 1 (Lines 8-12).
- Resampling: Particles with negligible weights are replaced by new particles that are copies of particles with higher weights (Line 13). This step mitigates the issue of particle degeneracy, where the weights become overly concentrated on a very small subset of particles (in an extreme case, only all by one particle have close-to-zero weights).

Appendix B: Decomposition of likelihood and Bayesian inference in a DBN

Using the DBN given in Fig. 16 as an example, the posterior distribution $p(x_{1,k}, x_{2,k}, x_{3,k}, x_{4,k}|y_{1,k})$ of state variables $x_{1,k}$, $x_{2,k}$, $x_{3,k}$, and $x_{4,k}$ for given observation $y_{1,k}$ at time t_k is given by

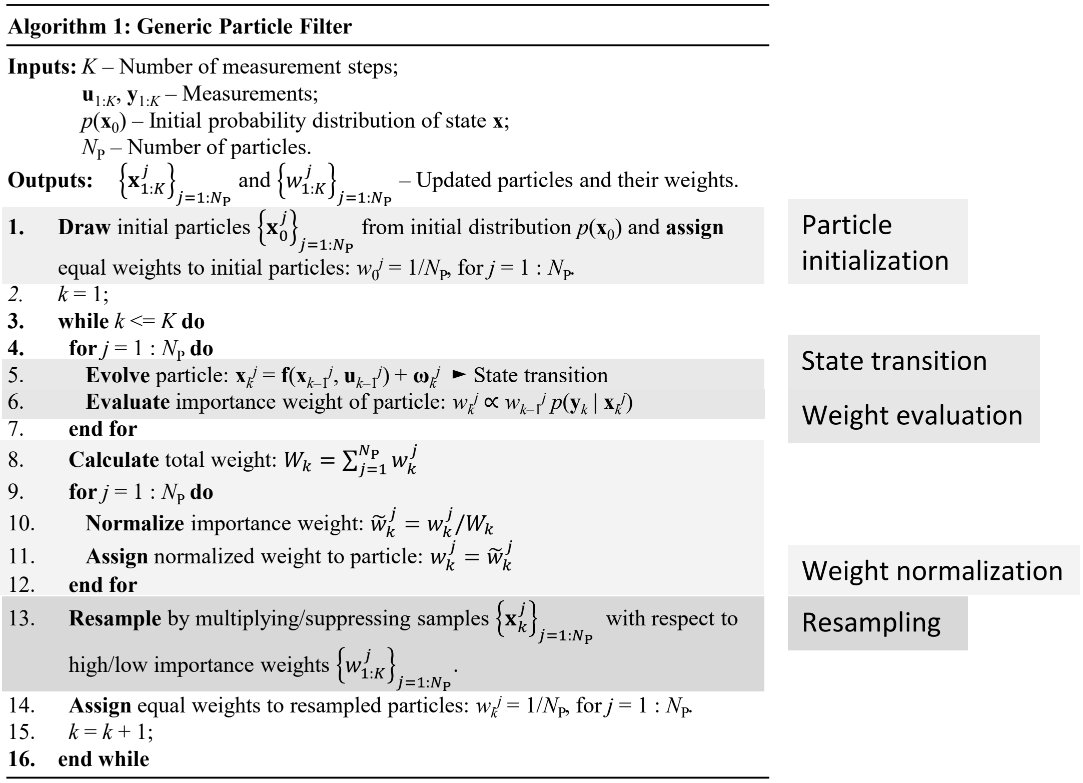
$$\begin{aligned} &p(x_{1,k}, x_{2,k}, x_{3,k}, x_{4,k}|y_{1,k}) \\ &\propto p(y_{1,k}|x_{3,k}, x_{4,k}) p(x_{3,k}|x_{2,k}, x_{1,k}) \\ &\times p(x_{4,k}|x_{2,k}, x_{1,k}) p'(x_{1,k}, x_{2,k}, x_{3,k}, x_{4,k}), \end{aligned} \tag{14}$$

where $p(y_{1,k}|x_{3,k}, x_{4,k})$, $p(x_{3,k}|x_{2,k}, x_{1,k})$, and $p(x_{4,k}|x_{2,k}, x_{1,k})$ are the conditional probability tables or conditional probability distributions describing the probabilistic causality relationship between the parent nodes (e.g., $x_{2,k}$ and $x_{1,k}$) and a child node (e.g., $x_{3,k}$), $p'(x_{1,k}, x_{2,k}, x_{3,k})$ is the prior distribution of state variables $x_{1,k}$, $x_{2,k}$, $x_{3,k}$, and $x_{4,k}$ at t_k, which is obtained at each time step by recursively performing Bayesian inference and uncertainty propagation based on observations and state transition probabilities given in Eq. (6).

Theoretically, all the Bayesian inference methods discussed in Sec. 4.2.2 can be employed to update the posterior of state variables based on the formulation given in Eq. (14). In the context of digital twins, particle filters such as the one presented in Appendix A are usually used in conjunction with DBNs to update digital states for two main reasons: (1) particle filters have fewer assumptions on the nonlinearity of the state transition and measurement functions and noise distributions than Kalman filters (see Table 4); and (2) using surrogate models to approximate conditional probability distributions (see Eq. (5)) allows for an efficient evaluation of likelihood functions, such that a large number of particles can be used for the inference. Using Eq. (14) as an example, in a particle filter implementation, we first generate equally weighted particles of state variables $x_{1,k}, x_{2,k}$, $x_{3,k}$, and $x_{4,k}$ as their prior at t_k according to the transition probabilities of state variables defined by the transition BN (i.e., Step 5 in Fig. 28). After that, the likelihood of each particle is computed using the decomposed likelihood function $p(y_{1,k}|x_{3,k}, x_{4,k})p(x_{3,k}|x_{2,k}, x_{1,k}) \times p(x_{4,k}|x_{2,k}, x_{1,k})$ given in Eq. (14). This corresponds to the computation of particle weights in Step 6 of Fig. 28. The likelihood values of the particles are then normalized and the particles are subsequently re-sampled based on the normalized weights to obtain the posterior samples of the state variables at t_k. This process is implemented repeatedly over time to dynamically update the state variables.

Acknowledgements

Adam Thelen and Chao Hu would like to thank the financial support from the U.S. National Science Foundation under Grant No. ECCS-2015710. Xiaoge Zhang is supported by a grant from the Research Committee of The Hong Kong Polytechnic University under project code 1-BE6V and G-UAMR. Sankaran Mahadevan acknowledges the support of the National Institute of Science and Technology. Michael D. Todd and Zhen Hu received financial support from the U.S. Army



```
Algorithm 1: Generic Particle Filter
Inputs: $K$ – Number of measurement steps;
        $\mathbf{u}_{1:K}$, $\mathbf{y}_{1:K}$ – Measurements;
        $p(\mathbf{x}_0)$ – Initial probability distribution of state $\mathbf{x}$;
        $N_\mathrm{P}$ – Number of particles.
Outputs: $\{\mathbf{x}_{1:K}^j\}_{j=1:N_\mathrm{P}}$ and $\{w_{1:K}^j\}_{j=1:N_\mathrm{P}}$ – Updated particles and their weights.
Draw initial particles $\{\mathbf{x}_0^j\}_{j=1:N_\mathrm{P}}$ from initial distribution $p(\mathbf{x}_0)$ and assign
    equal weights to initial particles: $w_0^j = 1/N_\mathrm{P}$, for $j = 1 : N_\mathrm{P}$.
$k = 1$;
while $k <= K$ do
  for $j = 1 : N_\mathrm{P}$ do
    Evolve particle: $\mathbf{x}_k^j = \mathbf{f}(\mathbf{x}_{k-1}^j, \mathbf{u}_{k-1}^j) + \boldsymbol{\omega}_k^j$  ► State transition
    Evaluate importance weight of particle: $w_k^j \propto w_{k-1}^j\, p(\mathbf{y}_k \mid \mathbf{x}_k^j)$
  end for
  Calculate total weight: $W_k = \sum_{j=1}^{N_P} w_k^j$
  for $j = 1 : N_\mathrm{P}$ do
    Normalize importance weight: $\tilde{w}_k^j = w_k^j / W_k$
    Assign normalized weight to particle: $w_k^j = \tilde{w}_k^j$
  end for
  Resample by multiplying/suppressing samples $\{\mathbf{x}_k^j\}_{j=1:N_\mathrm{P}}$ with respect to
    high/low importance weights $\{w_{1:K}^j\}_{j=1:N_\mathrm{P}}$.
  Assign equal weights to resampled particles: $w_k^j = 1/N_\mathrm{P}$, for $j = 1 : N_\mathrm{P}$.
  $k = k + 1$;
end while
```

Fig. 28: Pseudocode of a generic particle filter algorithm

Corps of Engineers through the U.S. Army Engineer Research and Development Center Research Cooperative Agreement W912HZ-17-2-0024.

Competing interests

The authors have no relevant financial or non-financial conflicts of interest to disclose.

Replication of results

The Python code and preprocessed dataset used for the battery case study are available for download on Github.

Authors' contributions

All authors read and approved the final manuscript. Hu, C. and Hu, Z. devised the original concept of the review paper. Hu, Z, Thelen, A., and Zhang, X. were responsible for the literature review. Thelen, A. was responsible for geometric modeling. Hu, C., Thelen, A., and Hu, Z. were responsible for physics-based modeling. Hu, Z. was responsible for data-driven modeling. Hu, C. and Hu, Z. were responsible for physics-informed ML. Zhang, X. was responsible for system modeling. Hu, C., and Hu, Z., were responsible for probabilistic model updating. Zhang, X. was responsible for ML model updating. Hu, C. and Hu., Z were responsible for fault diagnostics, failure prognostics, and predictive maintenance. Lu, Y. was responsible for MPC. Fink, O. was responsible for federated learning and domain adaptation. Zhang, X., Hu, Z., and Fink, O. were responsible for deep reinforcement learning. Hu, C. was responsible for UQ of ML models. Hu, Z. was responsible for UQ of dynamic system models, optimization for sensor placement, and optimization for physical system modeling. Lu, Y. was responsible for the optimization of additive manufacturing processes. Zhang, X. and Hu, Z. were responsible for real-time mission planning. Thelen, A. and Hu, C. were responsible for the case study and predictive maintenance scheduling. Hu, C. was responsible for open-source software and data. Ghosh, S. was responsible for the industry demonstration. Hu, C., Todd, M., and Mahadevan, S. were responsible for perspectives. All authors participated in manuscript writing, review, editing, and comment.

References

L. Aitchison. Bayesian filtering unifies adaptive and non-adaptive neural network optimization methods. *Advances in Neural Information Processing Systems*, 33:18173–18182, 2020.

J. Akroyd, S. Mosbach, A. Bhave, and M. Kraft. Universal digital twin-a dynamic knowledge graph. *Data-Centric Engineering*, 2, 2021.

S. Allu, S. Kalnaus, W. Elwasif, S. Simunovic, J. A. Turner, and S. Pannala. A new open computational framework for highly-resolved coupled three-dimensional multiphysics simulations of li-ion cells. *Journal of Power Sources*, 246:876–886, 2014.

A. Almalki, D. Downing, B. Lozanovski, R. Tino, A. Du Plessis, M. Qian, M. Brandt, and M. Leary. A digital-twin methodology for the non-destructive certification of lattice structures. *JOM*, 74(4):1784–1797, 2022.

G. Angjeliu, D. Coronelli, and G. Cardani. Development of the simulation model for digital twin applications in historical masonry buildings: The integration between numerical and experimental reality. *Computers & Structures*, 238: 106282, 2020.

I. ANSYS. Ansys and ptc partner to create a digital twin of an electric motor. https://www.ansys.com/resource-center/demo/ansys-ptc-demo. Last Accessed: 2022-05-20.

J. Antoni and R. B. Randall. The spectral kurtosis: application to the vibratory surveillance and diagnostics of rotating machines. *Mechanical Systems and Signal Processing*, 20(2):308–331, 2006.

P. D. Arendt, D. W. Apley, and W. Chen. Quantification of model uncertainty: Calibration, model discrepancy, and identifiability. 2012.

M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

K. J. Åström and P. Eykhoff. System identification—a survey. *Automatica*, 7(2):123–162, 1971.

R. Automation. Inca mpc: Model-predictive multi-variable controller. https://literature.rockwellautomation.com/idc/groups/literature/documents/br/rsbrp8-br001_-en-p.pdf. Last Accessed: 2022-05-29.

M. Ayani, M. Ganebäck, and A. H. Ng. Digital twin: Applying emulation for machine reconditioning. *Procedia CIRP*, 72:243–248, 2018.

E. Ayerbe, M. Berecibar, S. Clark, A. A. Franco, and J. Ruhland. Digitalization of battery manufacturing: Current status, challenges, and opportunities. *Advanced Energy Materials*, page 2102696, 2021.

M. Aykol, C. B. Gopal, A. Anapolsky, P. K. Herring, B. van Vlijmen, M. D. Berliner, M. Z. Bazant, R. D. Braatz, W. C. Chueh, and B. D. Storey. Perspective—combining physics and machine learning to predict battery lifetime. *Journal of The Electrochemical Society*, 168(3): 030525, 2021.

G. Bachelor, E. Brusa, D. Ferretto, and A. Mitschke. Model-based design of complex aeronautical systems through digital twin and thread concepts. *IEEE Systems Journal*, 14(2): 1568–1579, 2019.

A. Banerjee, R. Dalal, S. Mittal, and K. P. Joshi. Generating digital twin models using knowledge graphs for industrial production lines. *UMBC Information Systems Department*, 2017.

P. Bangalore and M. Patriksson. Analysis of scada data for early fault detection, with application to the maintenance management of wind turbines. *Renewable Energy*, 115:521–532, 2018.

B. R. Barricelli, E. Casiraghi, and D. Fogli. A survey on digital twin: Definitions, characteristics, applications, and design implications. *IEEE Access*, 7:167653–167671, 2019.

N. Barry, M. Chatzos, W. Chen, D. Han, C. Huang, R. Joseph, M. Klamkin, S. Park, M. Tanneau, P. Van Hentenryck, et al. Risk-aware control and optimization for high-renewable power grids. *arXiv preprint arXiv:2204.00950*, 2022.

F. Bellalouna. Case study for design optimization using the digital twin approach. *Procedia CIRP*, 100:595–600, 2021.

Y. Bengio, A. Lodi, and A. Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, 290(2):405–421, 2021.

M. Berecibar, I. Gandiaga, I. Villarreal, N. Omar, J. Van Mierlo, and P. Van den Bossche. Critical review of state of health estimation methods of li-ion batteries for real applications. *Renewable and Sustainable Energy Reviews*, 56:572–587, 2016.

B. J. Bichon, M. S. Eldred, L. P. Swiler, S. Mahadevan, and J. M. McFarland. Efficient global reliability analysis for nonlinear implicit performance functions. *AIAA journal*, 46(10): 2459–2468, 2008.

S. A. Billings. *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons, 2013.

J. Bilmes. Graphical models toolkit. https://web.archive.org/web/20150228005644/http://melodi.ee.washington.edu/gmtk/. Last Accessed: 2022-04-24.

C. R. Birkl, M. R. Roberts, E. McTurk, P. G. Bruce, and D. A. Howey. Degradation diagnostics for lithium ion cells. *Journal of Power Sources*, 341:373–386, 2017.

I. Bloom, B. Cole, J. Sohn, S. A. Jones, E. G. Polzin, V. S. Battaglia, G. L. Henriksen, C. Motloch, R. Richardson, T. Unkelhaeuser, et al. An accelerated calendar and cycle life study of li-ion cells. *Journal of power sources*, 101(2):238–247, 2001.

C. Boje, A. Guerriero, S. Kubicki, and Y. Rezgui. Towards a semantic construction digital twin: Directions for future research. *Automation in Construction*, 114:103179, 2020.

C. Boris, A. Sadek, and C. Wolf. Universal domain adaptation in ordinal regression. *arXiv preprint arXiv:2106.11576*, 2021.

K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4243–4250. IEEE, 2018.

G. E. Box. Science and statistics. *Journal of the American Statistical Association*, 71(356):791–799, 1976.

R. Brincker and P. Andersen. Understanding stochastic subspace identification. In *Conference Proceedings: IMAC-XXIV: A Conference & Exposition on Structural Dynamics*. Society for Experimental Mechanics, 2006.

R. Brincker, L. Zhang, and P. Andersen. Modal identification of output-only systems using frequency domain decomposition. *Smart Materials and Structures*, 10(3):441, 2001.

D. G. Broo, M. Bravo-Haro, and J. Schooling. Design and implementation of a smart infrastructure digital twin. *Automation in Construction*, 136:104171, 2022.

S. Brooks. Markov chain monte carlo method and its application. *Journal of the Royal Statistical Society: Series D (the Statistician)*, 47(1):69–100, 1998.

D. Bzdok, M. Krzywinski, and N. Altman. Machine learning: a primer. *Nature methods*, 14 (12):1119, 2017.

Y. Cai, Y. Wang, and M. Burnett. Using augmented reality to build digital twin for reconfigurable additive manufacturing system. *Journal of Manufacturing Systems*, 56:598–604, 2020.

O. Cappé, S. J. Godsill, and E. Moulines. An overview of existing methods and recent advances in sequential monte carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò. Autodial: Automatic domain alignment layers. In *ICCV*, pages 5077–5085, 2017.

P. Castells, M. Fernandez, and D. Vallet. An adaptation of the vector-space model for ontology-based information retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 19(2): 261–272, 2006.

Z. Chai and C. Zhao. Enhanced random forest with concurrent analysis of static and dynamic nodes for industrial fault classification. *IEEE Transactions on Industrial Informatics*, 16(1): 54–66, 2019.

S. Chakraborty and S. Adhikari. Machine learning based digital twin for dynamical systems with multiple time-scales. *Computers & Structures*, 243:106410, 2021.

S. Chakraborty, S. Adhikari, and R. Ganguli. The role of surrogate models in the development of digital twins of dynamic systems. *Applied Mathematical Modelling*, 90:662–681, 2021.

M. A. Chao, C. Kulkarni, K. Goebel, and O. Fink. Hybrid deep fault detection and isolation: Combining deep neural networks and system performance models. *International Journal of Prognostics and Health Management*, 10(4), 2019.

M. A. Chao, C. Kulkarni, K. Goebel, and O. Fink. Fusing physics-based and deep learning models for prognostics. *Reliability Engineering & System Safety*, 217:107961, 2022.

S. Chen and S. A. Billings. Neural networks for nonlinear dynamic system modelling and identification. *International journal of control*, 56 (2):319–346, 1992.

W. Chen, S. Park, M. Tanneau, and P. Van Hentenryck. Learning optimization proxies for large-scale security-constrained economic dispatch. *Electric Power Systems Research*, 213: 108566, 2022.

X. Chen, S. Jia, and Y. Xiang. A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications*, 141:112948, 2020a.

Y. Chen, F. Luo, T. Li, T. Xiang, Z. Liu, and J. Li. A training-integrity privacy-preserving federated learning scheme with trusted execution environment. *Information Sciences*, 522: 69–79, 2020b.

Y.-J. Chen. Development of a method for ontology-based empirical knowledge representation and reasoning. *Decision Support Systems*, 50(1):1–20, 2010.

J. Cheng, W. Chen, F. Tao, and C.-L. Lin. Industrial iot in 5g environment towards smart manufacturing. *Journal of Industrial Information Integration*, 10:10–19, 2018.

J. Chiachío, M. L. Jalon, M. Chiachío, and A. Kolios. A markov chains prognostics framework for complex degradation processes. *Reliability Engineering & System Safety*, 195:106621, 2020.

S. H. Choi, K.-B. Park, D. H. Roh, J. Y. Lee, M. Mohammed, Y. Ghasemi, and H. Jeong. An integrated mixed reality system for safety-aware human-robot collaboration using deep learning and digital twin generation. *Robotics and Computer-Integrated Manufacturing*, 73: 102258, 2022.

A. J. Chorin, J. E. Marsden, and J. E. Marsden. *A mathematical introduction to fluid mechanics*, volume 3. Springer, 1990.

C. Cimino, E. Negri, and L. Fumagalli. Review of digital twin applications in manufacturing. *Computers in Industry*, 113:103130, 2019.

D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

M. Compare, F. Martini, S. Mattafirri, F. Carlevaro, and E. Zio. Semi-markov model for the oxidation degradation mechanism in gas turbine nozzles. *IEEE Transactions on Reliability*, 65 (2):574–581, 2016.

M. Compton, P. Barnaghi, L. Bermudez, R. Garcia-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog, et al. The ssn ontology of the w3c semantic sensor network incubator group. *Journal of Web Semantics*, 17:25–32, 2012.

J. Corral-Acero, F. Margara, M. Marciniak, C. Rodero, F. Loncaric, Y. Feng, A. Gilbert, J. F. Fernandes, H. A. Bukhari, A. Wajdan, et al. The 'digital twin'to enable the vision of precision cardiology. *European Heart Journal*, 41(48):4556–4564, 2020.

H. Cox. On the estimation of state variables and parameters for noisy dynamic systems. *IEEE Transactions on Automatic Control*, 9(1):5–12, 1964.

A. J. Culler and J. J. McNamara. Studies on fluid-thermal-structural coupling for aerothermoelasticity in hypersonic flow. *AIAA journal*, 48(8):1721–1738, 2010.

P. R. d. O. da Costa, A. Akçay, Y. Zhang, and U. Kaymak. Remaining useful lifetime prediction via deep domain adaptation. *Reliability Engineering & System Safety*, 195:106682, 2020.

Y. Dai, K. Zhang, S. Maharjan, and Y. Zhang. Deep reinforcement learning for stochastic computation offloading in digital twin networks. *IEEE Transactions on Industrial Informatics*, 17(7):4968–4977, 2020.

B. Danilo, A. Naomi, and K. Martin. Statistics versus machine learning. *Nature Methods*, 15 (4):233, 2018.

S. E. Davis, T. A. Lasko, G. Chen, E. D. Siew, and M. E. Matheny. Calibration drift in regression and machine learning models for acute kidney injury. *Journal of the American Medical Informatics Association*, 24(6):1052–1061, 2017.

D. A. DeLaurentis and D. Mavris. Uncertainty modeling and management in multidisciplinary analysis and synthesis. In *38th Aerospace Sciences Meeting and Exhibit*, page 422, 2000.

K. Deng, Y. Sun, S. Li, Y. Lu, J. Brouwer, P. G. Mehta, M. Zhou, and A. Chakraborty. Model predictive control of central chiller plant with thermal energy storage via dynamic programming and mixed-integer linear programming. *IEEE Transactions on Automation Science and Engineering*, 12:565–579, 2014.

T. Deng, K. Zhang, and Z.-J. M. Shen. A systematic review of a digital twin city: A new pattern of urban governance toward smart cities. *Journal of Management Science and Engineering*, 6 (2):125–134, 2021.

K. Ding, F. T. Chan, X. Zhang, G. Zhou, and F. Zhang. Defining a digital twin-based cyber-physical production system for autonomous manufacturing in smart shop floors. *International Journal of Production Research*, 57(20): 6315–6334, 2019.

A. Doerr, C. Daniel, M. Schiegg, N.-T. Duy, S. Schaal, M. Toussaint, and T. Sebastian. Probabilistic recurrent state-space models. In *International Conference on Machine Learning*, pages 1280–1289. PMLR, 2018.

M. Döhler, E. Reynders, F. Magalhaes, L. Mevel, G. D. Roeck, and A. Cunha. Pre-and post-identification merging for multi-setup oma with covariance-driven ssi. In *Dynamics of Bridges, Volume 5*, pages 57–70. Springer, 2011.

A. Downey, Y.-H. Lui, C. Hu, S. Laflamme, and S. Hu. Physics-based prognostics of lithium-ion battery using non-linear least squares with dynamic bounds. *Reliability Engineering & System Safety*, 182:1–12, 2019.

K. Dröder, P. Bobka, T. Germann, F. Gabriel, and F. Dietrich. A machine learning-enhanced digital twin approach for human-robot-collaboration. *Procedia Cirp*, 76:187–192, 2018.

H. Elayan, M. Aloqaily, and M. Guizani. Digital twin for intelligent context-aware iot healthcare systems. *IEEE Internet of Things Journal*, 8 (23):16749–16757, 2021.

M. J. Ellis. Economic model predictive control theory: Computational efficiency and application to smart manufacturing. 2015.

J. A. Erkoyuncu, I. F. del Amo, D. Ariansyah, D. Bulka, R. Roy, et al. A design framework for adaptive digital twins. *CIRP annals*, 69(1): 145–148, 2020.

I. Errandonea, S. Beltrán, and S. Arrizabalaga. Digital twin for maintenance: A literature review. *Computers in Industry*, 123:103316, 2020.

A. Essien and C. Giannetti. A deep learning model for smart manufacturing using convolutional lstm neural network autoencoders. *IEEE Transactions on Industrial Informatics*, 16:6069, 2021.

L. Fan, J. Khodadadi, and A. Pesaran. A parametric study on thermal management of an air-cooled lithium-ion battery module for plug-in hybrid electric vehicles. *Journal of Power Sources*, 238:301–312, 2013.

W. Fan and P. Qiao. Vibration-based damage identification methods: a review and comparative study. *Structural health monitoring*, 10(1): 83–111, 2011.

Y. Fang, C. Peng, P. Lou, Z. Zhou, J. Hu, and J. Yan. Digital-twin-based job shop scheduling toward smart manufacturing. *IEEE Transactions on Industrial Informatics*, 15(12):6425–6435, 2019.

C. R. Farrar and K. Worden. An introduction to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365 (1851):303–315, 2007.

F. Farzan, M. A. Jafari, R. Masiello, and Y. Lu. Toward optimal day-ahead scheduling and operation control of microgrids under uncertainty. *IEEE Transactions on Smart Grid*, 6:499–507, 2014.

M. G. Fernández-Godino, C. Park, N.-H. Kim, and R. T. Haftka. Review of multi-fidelity models. *arXiv preprint arXiv:1609.07196*, 2016.

O. Fink, E. Zio, and U. Weidmann. Predicting component reliability and level of degradation with complex-valued neural networks. *Reliability Engineering & System Safety*, 121:198–206, 2014.

O. Fink, Q. Wang, M. Svensen, P. Dersin, W.-J. Lee, and M. Ducoffe. Potential, challenges and future directions for deep learning in prognostics and health management applications. *Engineering Applications of Artificial Intelligence*, 92:103678, 2020.

P. Franciosa, M. Sokolov, S. Sinha, T. Sun, and D. Ceglarek. Deep learning enhanced digital twin for closed-loop in-process quality improvement. *CIRP Annals*, 69(1):369–372, 2020.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2):131–163, 1997.

A. Fuller, Z. Fan, C. Day, and C. Barlow. Digital twin: Enabling technologies, challenges and open research. *IEEE Access*, 8:108952–108971, 2020.

Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pages 1180–1189. JMLR. org, 2015.

Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

H. Gao, L. Sun, and J.-X. Wang. Phygeonet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state pdes on irregular domain. *Journal of Computational Physics*, 428:110079, 2021.

P. Gasper, K. Gering, E. Dufek, and K. Smith. Challenging practices of algebraic battery life models through statistical validation and model identification via machine-learning. *Journal of the Electrochemical Society*, 168(2):020502, 2021.

N. Z. Gebraeel, M. A. Lawley, R. Li, and J. K. Ryan. Residual-life distributions from component degradation signals: A bayesian approach. *IIE Transactions*, 37(6):543–557, 2005.

M. Giorgio, M. Guida, and G. Pulcini. An age-and state-dependent markov model for degradation processes. *IIE Transactions*, 43(9):621–632, 2011.

M. Giselle Fernández-Godino, C. Park, N. H. Kim, and R. T. Haftka. Issues in deciding whether to use multifidelity surrogates. *AIAA Journal*, 57 (5):2039–2054, 2019.

Github. Link to github repository where the preprocessed data and python scripts used to generate all the results and figures in the case study section reside. https://github.com/acthelen/battery_digital_twin.

E. Glaessgen and D. Stargel. The digital twin paradigm for future nasa and us air force vehicles. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, page 1818, 2012.

M. Glatt, C. Sinnwell, L. Yi, S. Donohoe, B. Ravani, and J. C. Aurich. Modeling and implementation of a digital twin of material flows based on physics simulation. *Journal of Manufacturing Systems*, 58:231–245, 2021.

T. Goodwin, J. Xu, N. Celik, and C.-H. Chen. Real-time digital twin-based optimization with predictive simulation learning. *Journal of Simulation*, pages 1–18, 2022.

N. Gorjian, L. Ma, M. Mittinty, P. Yarlagadda, and Y. Sun. A review on degradation models in reliability analysis. In *Engineering asset lifecycle management*, pages 369–384. Springer, 2010.

M. Grieves. Digital twin: manufacturing excellence through virtual factory replication. *White paper*, 1:1–7, 2014.

P. Guillaume, P. Verboven, S. Vanlanduit, H. Van Der Auweraer, and B. Peeters. A poly-reference implementation of the least-squares complex frequency-domain estimator. In *Proceedings of IMAC*, volume 21, pages 183–192. A Conference & Exposition on Structural Dynamics, Society for Experimental . . . , 2003.

D. Guivarch, E. Mermoz, Y. Marino, and M. Sartor. Creation of helicopter dynamic systems digital twin using multibody simulations. *CIRP Annals*, 68(1):133–136, 2019.

H. Guo, M. Chen, K. Mohamed, T. Qu, S. Wang, and J. Li. A digital twin-based flexible cellular manufacturing for optimization of air conditioner line. *Journal of Manufacturing Systems*, 58:65–78, 2021.

L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin. A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing*, 240:98–109, 2017.

S. Guo, W. Guo, L. Bian, and Y. Guo. A deep-learning-based surrogate model for thermal signature prediction in laser metal deposition. *IEEE Transactions on Automation Science and Engineering*, 2022.

F. Gustafsson. Particle filter theory and practice with positioning applications. *IEEE Aerospace and Electronic Systems Magazine*, 25(7):53–82, 2010.

S. Ha, S. Coros, A. Alspach, J. Kim, and K. Yamane. Computational co-optimization of design parameters and motion trajectories for robotic systems. *The International Journal of*

Robotics Research, 37(13-14):1521–1536, 2018.

S. Haag and R. Anderl. Digital twin–proof of concept. *Manufacturing letters*, 15:64–66, 2018.

E. Haghighat, M. Raissi, A. Moure, H. Gomez, and R. Juanes. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 379: 113741, 2021.

B. He and K.-J. Bai. Digital twin-based sustainable intelligent manufacturing: A review. *Advances in Manufacturing*, 9(1):1–21, 2021.

R. He, G. Chen, C. Dong, S. Sun, and X. Shen. Data-driven digital twin technology for optimized control in process systems. *ISA Transactions*, 95:221–234, 2019.

W. He, N. Williard, M. Osterman, and M. Pecht. Prognostics of lithium-ion batteries based on dempster–shafer theory and the bayesian monte carlo method. *Journal of Power Sources*, 196 (23):10314–10321, 2011.

D. Heckerman. A tutorial on learning with bayesian networks. *Innovations in Bayesian networks*, pages 33–82, 2008.

A. Heng, S. Zhang, A. C. Tan, and J. Mathew. Rotating machinery prognostics: State of the art, challenges and opportunities. *Mechanical Systems and Signal Processing*, 23(3):724–739, 2009.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

M.-C. Hsu and Y. Bazilevs. Fluid–structure interaction modeling of wind turbines: simulating the full machine. *Computational Mechanics*, 50 (6):821–833, 2012.

C. Hu and B. D. Youn. Adaptive-sparse polynomial chaos expansion for reliability analysis and design of complex engineering systems. *Structural and Multidisciplinary Optimization*, 43(3): 419–442, 2011.

C. Hu, B. D. Youn, and J. Chung. A multiscale framework with extended Kalman filter for lithium-ion battery SOC and capacity estimation. *Applied Energy*, 92:694–704, 2012.

C. Hu, H. Ye, G. Jain, and C. Schmidt. Remaining useful life assessment of lithium-ion batteries in implantable medical devices. *Journal of Power Sources*, 375:118–130, 2018.

W. Hu, Y. He, Z. Liu, J. Tan, M. Yang, and J. Chen. Toward a digital twin: Time series prediction based on a hybrid ensemble empirical mode decomposition and bo-lstm neural networks. *Journal of Mechanical Design*, 143(5), 2021a.

W. Hu, T. Zhang, X. Deng, Z. Liu, and J. Tan. Digital twin: A state-of-the-art review of its enabling technologies, applications and challenges. *Journal of Intelligent Manufacturing and Special Equipment*, 2021b.

W. Hu, J. Shao, Q. Jiao, C. Wang, J. Cheng, Z. Liu, and J. Tan. A new differentiable architecture search method for optimizing convolutional neural networks in the digital twin of intelligent robotic grasping. *Journal of Intelligent Manufacturing*, pages 1–19, 2022a.

W. Hu, C. Wang, F. Liu, X. Peng, P. Sun, and J. Tan. A grasps-generation-and-selection convolutional neural network for a digital twin of intelligent robotic grasping. *Robotics and Computer-Integrated Manufacturing*, 77: 102371, 2022b.

X. Hu, L. Xu, X. Lin, and M. Pecht. Battery lifetime prognostics. *Joule*, 4(2):310–346, 2020.

Z. Hu and S. Mahadevan. Adaptive surrogate modeling for time-dependent multidisciplinary reliability analysis. *Journal of Mechanical Design*, 140(2), 2018a.

Z. Hu and S. Mahadevan. Bayesian network learning for data-driven design. *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg*, 4 (4), 2018b.

Z. Hu and Z. P. Mourelatos. A sequential accelerated life testing framework for system reliability assessment with untestable components. *Journal of Mechanical Design*, 140(10), 2018.

C.-G. Huang, H.-Z. Huang, and Y.-F. Li. A bidirectional lstm prognostics method under multiple operational conditions. *IEEE Transactions on Industrial Electronics*, 66(11):8792–8802, 2019.

R. Huang, L. Xi, X. Li, C. R. Liu, H. Qiu, and J. Lee. Residual life predictions for ball bearings based on self-organizing map and back propagation neural network methods. *Mechanical Systems and Signal Processing*, 21(1):193–207, 2007.

S. Huang, G. Wang, and Y. Yan. Building blocks for digital twin of reconfigurable machine tools from design perspective. *International Journal* *of Production Research*, 60(3):942–956, 2022a.

X. Huang, T. Xie, Z. Wang, L. Chen, Q. Zhou, and Z. Hu. A transfer learning-based multi-fidelity point-cloud neural network approach for melt pool modeling in additive manufacturing. *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg*, 8(1), 2022b.

M. S. Ibrahim, J. Fan, W. K. Yung, A. Prisacaru, W. van Driel, X. Fan, and G. Zhang. Machine learning and digital twin driven diagnostics and prognostics of light-emitting diodes. *Laser & Photonics Reviews*, 14(12):2000254, 2020.

S. Im, J. Lee, and M. Cho. Surrogate modeling of elasto-plastic problems via long short-term memory neural networks and proper orthogonal decomposition. *Computer Methods in Applied Mechanics and Engineering*, 385:114030, 2021.

S. Industry. Model predictive control - rockwell model predictive control delivers results. https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/10017369. Last Accessed: 2022-05-29.

A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics*, 404:109136, 2020a.

A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020b.

P. Jain, J. Poon, J. P. Singh, C. Spanos, S. R. Sanders, and S. K. Panda. A digital twin approach for fault diagnosis in distributed photovoltaic systems. *IEEE Transactions on Power Electronics*, 35(1):940–956, 2019.

A. K. Jardine, D. Lin, and D. Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7):1483–1510, 2006.

R. Jegadeeshwaran and V. Sugumaran. Fault diagnosis of automobile hydraulic brake system using statistical features and support vector machines. *Mechanical Systems and Signal Processing*, 52:436–446, 2015.

S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

C. Jiang, M. A. Vega, M. D. Todd, and Z. Hu. Model correction and updating of a stochastic degradation model for failure prognostics of miter gates. *Reliability Engineering & System Safety*, 218:108203, 2022.

F. Jiang, L. Ma, T. Broyd, and K. Chen. Digital twin and its implementations in the civil engineering sector. *Automation in Construction*, 130:103838, 2021.

D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks. Characterising the digital twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology*, 29: 36–52, 2020.

J.-N. Juang and R. S. Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *Journal of Guidance, Control, and Dynamics*, 8(5):620–627, 1985.

M. G. Kapteyn, D. J. Knezevic, D. Huynh, M. Tran, and K. E. Willcox. Data-driven physics-based digital twins via a library of component-based reduced-order models. *International Journal for Numerical Methods in Engineering*, 2020.

M. G. Kapteyn, J. V. Pretorius, and K. E. Willcox. A probabilistic graphical model foundation for enabling predictive digital twins at scale. *Nature Computational Science*, 1(5):337–347, 2021.

B. Kapusuzoglu and S. Mahadevan. Physics-informed and hybrid machine learning in additive manufacturing: application to fused filament fabrication. *Jom*, 72(12):4695–4705, 2020.

G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

P. M. Karve, Y. Guo, B. Kapusuzoglu, S. Mahadevan, and M. A. Haile. Digital twin approach for damage-tolerant mission planning under uncertainty. *Engineering Fracture Mechanics*, 225: 106766, 2020.

M. C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

G. Kerschen, K. Worden, A. F. Vakakis, and J.-C. Golinval. Past, present and future of nonlinear system identification in structural dynamics.

Mechanical Systems and Signal Processing, 20 (3):505–592, 2006.

S. Khan and T. Yairi. A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing*, 107: 241–265, 2018.

S. Khan, M. Farnsworth, R. McWilliam, and J. Erkoyuncu. On the requirements of digital twin-driven autonomous maintenance. *Annual Reviews in Control*, 50:13–28, 2020.

J. P. Kharoufeh, C. J. Solo, and M. Y. Ulukus. Semi-markov models for degradation-based reliability. *IIE Transactions*, 42(8):599–612, 2010.

Y. Kim, K. Na, and B. D. Youn. A health-adaptive time-scale representation (htsr) embedded convolutional neural network for gearbox fault diagnostics. *Mechanical Systems and Signal Processing*, 167:108575, 2022.

S. Kohtz, Y. Xu, Z. Zheng, and P. Wang. Physics-informed machine learning model for battery state of health prognostics using partial charging segments. *Mechanical Systems and Signal Processing*, 172:109002, 2022.

W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn. Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, 2018.

A. Ladj, Z. Wang, O. Meski, F. Belkadi, M. Ritou, and C. Da Cunha. A knowledge-based digital shadow for machining industry in a digital twin perspective. *Journal of Manufacturing Systems*, 58:168–179, 2021.

N. Lao and W. W. Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine learning*, 81(1):53–67, 2010.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

C.-Y. Lee, T. Batra, M. H. Baig, and D. Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10285–10295, 2019.

J. Lee, E. Lapira, B. Bagheri, and H.-a. Kao. Recent advances and trends in predictive manufacturing systems in big data environment. *Manufacturing letters*, 1(1):38–41, 2013a.

J. Lee, E. Lapira, S. Yang, and A. Kao. Predictive manufacturing system-trends of next-generation production systems. *IFAC proceedings volumes*, 46(7):150–156, 2013b.

J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel. Prognostics and health management design for rotary machinery systems—reviews, methodology and applications. *Mechanical Systems and Signal Processing*, 42(1-2):314–334, 2014.

K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in Neural Information Processing Systems*, 31, 2018.

S. Lee, L. Li, and J. Ni. Online degradation assessment and adaptive fault detection using modified hidden markov model. *Journal of Manufacturing Science and Engineering*, 132 (2), 2010.

Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin. Machinery health prognostics: A systematic review from data acquisition to rul prediction. *Mechanical Systems and Signal Processing*, 104: 799–834, 2018.

J. Leng, H. Zhang, D. Yan, Q. Liu, X. Chen, and D. Zhang. Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop. *Journal of Ambient Intelligence and Humanized Computing*, 10(3): 1155–1166, 2019.

J. Leng, Q. Liu, S. Ye, J. Jing, Y. Wang, C. Zhang, D. Zhang, and X. Chen. Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model. *Robotics and Computer-Integrated Manufacturing*, 63:101895, 2020.

I. Lenz, R. A. Knepper, and A. Saxena. Deepmpc: Learning deep latent features for model predictive control. In *Robotics: Science and Systems*, 2015.

P. E. Leser, J. E. Warner, W. P. Leser, G. F. Bomarito, J. A. Newman, and J. D. Hochhalter. A digital twin feasibility study (part ii): Non-deterministic predictions of fatigue life using in-situ diagnostics and prognostics. *Engineering Fracture Mechanics*, 229:106903, 2020.

C. Li and G. H. Lee. From synthetic to real: Unsupervised domain adaptation for animal pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

Recognition, pages 1482–1491, 2021.

C. Li, S. Mahadevan, Y. Ling, S. Choze, and L. Wang. Dynamic bayesian network for aircraft wing health monitoring digital twin. *Aiaa Journal*, 55(3):930–941, 2017.

N. Li, Y. Lei, J. Lin, and S. X. Ding. An improved exponential model for predicting remaining useful life of rolling element bearings. *IEEE Transactions on Industrial Electronics*, 62(12): 7762–7773, 2015.

S. Li and Y. Yang. Hierarchical deep learning for data-driven identification of reduced-order models of nonlinear dynamical systems. *Nonlinear Dynamics*, 105(4):3409–3422, 2021.

T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020a.

W. Li, M. Rentemeister, J. Badeda, D. Jöst, D. Schulte, and D. U. Sauer. Digital twin for battery systems: Cloud battery management system with online state-of-charge and state-of-health estimation. *Journal of Energy Storage*, 30:101557, 2020b.

W. Li, R. Huang, J. Li, Y. Liao, Z. Chen, G. He, R. Yan, and K. Gryllias. A perspective survey on deep transfer learning for fault diagnosis in industrial scenarios: Theories, applications and challenges. *Mechanical Systems and Signal Processing*, 167:108487, 2022.

X. Li and W. Zhang. Deep learning-based partial domain adaptation method on intelligent machinery fault diagnostics. *IEEE Transactions on Industrial Electronics*, 68(5):4351–4361, 2020.

X. Li, Q. Ding, and J.-Q. Sun. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, 172:1–11, 2018.

X. Li, W. Zhang, and Q. Ding. Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction. *Reliability Engineering & System Safety*, 182:208–218, 2019a.

X. Li, W. Zhang, H. Ma, Z. Luo, and X. Li. Partial transfer learning in machinery cross-domain fault diagnostics using class-weighted adversarial networks. *Neural Networks*, 129:313–322, 2020c.

Y. Li, K. Liu, A. M. Foley, A. Zülke, M. Berecibar, E. Nanini-Maury, J. Van Mierlo, and H. E. Hoster. Data-driven health estimation and lifetime prediction of lithium-ion batteries: A review. *Renewable and Sustainable Energy Reviews*, 113:109254, 2019b.

Y. Li, D. M. Vilathgamuwa, E. Wikner, Z. Wei, X. Zhang, T. Thiringer, T. Wik, and C. Zou. Electrochemical model-based fast charging: Physical constraint-triggered pi control. *IEEE Transactions on Energy Conversion*, 36(4): 3208–3220, 2021.

Z. Li, Y. Jiang, C. Hu, and Z. Peng. Recent progress on decoupling diagnosis of hybrid failures in gear transmission systems using vibration sensor signal: A review. *Measurement*, 90: 4–19, 2016.

G. Liao, H. Yin, M. Chen, and Z. Lin. Remaining useful life prediction for multi-phase deteriorating process based on wiener process. *Reliability Engineering & System Safety*, 207:107361, 2021.

K. Y. H. Lim, P. Zheng, and C.-H. Chen. A state-of-the-art survey of digital twin: techniques, engineering product lifecycle management and business innovation perspectives. *Journal of Intelligent Manufacturing*, 31(6): 1313–1337, 2020a.

K. Y. H. Lim, P. Zheng, C.-H. Chen, and L. Huang. A digital twin-enhanced system for engineering product family design and optimization. *Journal of Manufacturing Systems*, 57:82–93, 2020b.

T. Lin, B. G. Horne, P. Tino, and C. L. Giles. Learning long-term dependencies in narx recurrent neural networks. *IEEE Transactions on Neural Networks*, 7(6):1329–1338, 1996.

Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI Conference on Artificial Intelligence*, 2015.

C. Liu, L. Le Roux, C. Körner, O. Tabaste, F. Lacan, and S. Bigot. Digital twin-enabled collaborative data management for metal additive manufacturing systems. *Journal of Manufacturing Systems*, 2020a.

C. Liu, A. Ricardo Mauricio, J. Qi, D. Peng, and K. Gryllias. Domain adaptation digital twin for rolling element bearing prognostics. *Online Proceedings of PHM2020*, pages 1–10, 2020b.

J. Liu, J. Liu, C. Zhuang, Z. Liu, and T. Miao. Construction method of shop-floor digital twin based on mbse. *Journal of Manufacturing Systems*, 60:93–118, 2021a.

L. Liu, X. Zhang, X. Wan, S. Zhou, and Z. Gao. Digital twin-driven surface roughness prediction and process parameter adaptive optimization. *Advanced Engineering Informatics*, 51:101470, 2022.

M. Liu, S. Fang, H. Dong, and C. Xu. Review of digital twin about concepts, technologies, and industrial applications. *Journal of Manufacturing Systems*, 58:346–361, 2021b.

Q. Liu, J. Leng, D. Yan, D. Zhang, L. Wei, A. Yu, R. Zhao, H. Zhang, and X. Chen. Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system. *Journal of Manufacturing Systems*, 58:52–64, 2021c.

S. Liu, Y. Lu, J. Li, D. Song, X. Sun, and J. Bao. Multi-scale evolution mechanism and knowledge construction of a digital twin mimic model. *Robotics and Computer-Integrated Manufacturing*, 71:102123, 2021d.

T. Liu, K. Zhu, and L. Zeng. Diagnosis and prognosis of degradation process via hidden semi-markov model. *IEEE/ASME Transactions on Mechatronics*, 23(3):1456–1466, 2018a.

X. Liu, B. T. Du Jiang, G. Jiang, Y. Sun, J. Kong, X. Tong, G. Zhao, and B. Chen. Genetic algorithm-based trajectory optimization for digital twin robots. *Frontiers in Bioengineering and Biotechnology*, 9, 2021e.

Y. Liu, L. Zhang, Y. Yang, L. Zhou, L. Ren, F. Wang, R. Liu, Z. Pang, and M. J. Deen. A novel cloud-based framework for the elderly healthcare services using digital twin. *IEEE Access*, 7:49088–49101, 2019.

Z. Liu, N. Meyendorf, and N. Mrad. The role of data fusion in predictive maintenance using digital twin. In *AIP Conference Proceedings*, volume 1949, page 020023. AIP Publishing LLC, 2018b.

Z. Liu, W. Chen, C. Zhang, C. Yang, and Q. Cheng. Intelligent scheduling of a feature-process-machine tool supernetwork based on digital twin workshop. *Journal of Manufacturing Systems*, 58:157–167, 2021f.

L. Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *IEEE Transactions on Automatic Control*, 24(1):36–50, 1979.

L. Ljung. System identification. In *Signal analysis and prediction*, pages 163–173. Springer, 1998.

L. Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.

L. Ljung and K. Glover. Frequency domain versus time domain methods in system identification. *Automatica*, 17(1):71–86, 1981.

A. Löcklin, M. Müller, T. Jung, N. Jazdi, D. White, and M. Weyrich. Digital twin for verification and validation of industrial automation systems–a survey. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 851–858. IEEE, 2020.

M. Lohtander, N. Ahonen, M. Lanz, J. Ratava, and J. Kaakkunen. Micro manufacturing unit and the corresponding 3d-model for the digital twin. *Procedia manufacturing*, 25:55–61, 2018.

M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.

J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363, 2018.

L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM Review*, 63(1): 208–228, 2021.

Q. Lu, A. K. Parlikad, P. Woodall, G. Don Ranasinghe, X. Xie, Z. Liang, E. Konstantinou, J. Heaton, and J. Schooling. Developing a digital twin at building and city levels: Case study of west cambridge campus. *Journal of Management in Engineering*, 36(3):05020004, 2020a.

Q. Lu, X. Xie, A. K. Parlikad, and J. M. Schooling. Digital twin-enabled anomaly detection for built asset monitoring in operation and maintenance. *Automation in Construction*, 118: 103277, 2020b.

Y. Lu, K. Morris, and S. Frechett. Current standards landscape for smart manufacturing systems. *NISTIR*, 8107:1–39, 2016.

Y. H. Lui, M. Li, A. Downey, S. Shen, V. P. Nemani, H. Ye, C. VanElzen, G. Jain, S. Hu, S. Laflamme, et al. Physics-based prognostics of implantable-grade lithium-ion battery for remaining useful life prediction. *Journal of Power Sources*, 485:229327, 2021.

W. Luo, T. Hu, C. Zhang, and Y. Wei. Digital twin for cnc machine tool: modeling and using strategy. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1129–1140, 2019.

W. Luo, T. Hu, Y. Ye, C. Zhang, and Y. Wei. A hybrid predictive maintenance approach for cnc machine tool driven by digital twin. *Robotics and Computer-Integrated Manufacturing*, 65: 101974, 2020.

G. P. Lydon, S. Caranovic, I. Hischier, and A. Schlueter. Coupled simulation of thermally active building systems to support a digital twin. *Energy and Buildings*, 202:109298, 2019.

J. P. Lynch and K. J. Loh. A summary review of wireless sensors and sensor networks for structural health monitoring. *Shock and Vibration Digest*, 38(2):91–130, 2006.

C. Ma, H. Gui, and J. Liu. Self learning-empowered thermal error control method of precision machine tools based on digital twin. *Journal of Intelligent Manufacturing*, pages 1–23, 2021.

P. Malhotra, V. Tv, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff. Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder. *arXiv preprint arXiv:1608.06154*, 2016.

S. G. Marquis, R. Timms, V. Sulzer, C. P. Please, and S. J. Chapman. A suite of reduced-order models of a single-layer lithium-ion pouch cell. *Journal of The Electrochemical Society*, 167 (14):140513, 2020.

M. Martínez-Ramón, J. L. Rojo-Alvarez, G. Camps-Valls, J. Muñoz-Marí, E. Soria-Olivas, A. R. Figueiras-Vidal, et al. Support vector machines for nonlinear kernel arma system identification. *IEEE Transactions on Neural Networks*, 17(6):1617–1622, 2006.

D. Masti and A. Bemporad. Learning nonlinear state–space models using autoencoders. *Automatica*, 129:109666, 2021.

M. Matulis and C. Harvey. A robot arm digital twin utilising reinforcement learning. *Computers & Graphics*, 95:106–114, 2021.

B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics (AISTATS)*, 2017.

H. Millwater, J. Ocampo, and N. Crosby. Probabilistic methods for risk assessment of airframe digital twin structures. *Engineering Fracture Mechanics*, 221:106674, 2019.

Q. Min, Y. Lu, Z. Liu, C. Su, and B. Wang. Machine learning based digital twin framework for production optimization in petrochemical industry. *International Journal of Information Management*, 49:502–519, 2019.

M. Misra, H. H. Yue, S. J. Qin, and C. Ling. Multivariate process monitoring and fault diagnosis by multi-scale pca. *Computers & Chemical Engineering*, 26(9):1281–1293, 2002.

T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, et al. Never-ending learning. *Communications of the ACM*, 61(5): 103–115, 2018.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

T. Moi, A. Cibicik, and T. Rølvåg. Digital twin based condition monitoring of a knuckle boom crane: An experimental study. *Engineering Failure Analysis*, 112:104517, 2020.

R. Molinaro, J.-S. Singh, S. Catsoulis, C. Narayanan, and D. Lakehal. Embedding data analytics and cfd into the digital twin concept. *Computers & Fluids*, 214:104759, 2021.

D. Montes de Oca Zapiain, J. A. Stewart, and R. Dingreville. Accelerating phase-field-based microstructure evolution predictions via surrogate models trained by machine learning methods. *npj Computational Materials*, 7(1):1–11, 2021.

K. Moustafa, Z. Hu, Z. P. Mourelatos, I. Baseski, and M. Majcher. System reliability analysis using component-level and system-level accelerated life testing. *Reliability Engineering & System Safety*, 214:107755, 2021.

K. P. Murphy. *Dynamic bayesian networks: representation, inference and learning*. University of California, Berkeley, 2002.

A. Neelakantan, B. Roth, and A. McCallum. Compositional vector space models for knowledge base inference. In *2015 AAAI Spring Symposium Series*, 2015.

V. P. Nemani, H. Lu, A. Thelen, C. Hu, and A. T. Zimmerman. Ensembles of probabilistic lstm predictors and correctors for bearing prognostics using industrial standards. *Neurocomputing*, 2021.

H. X. Nguyen, R. Trestian, D. To, and M. Tatipamula. Digital twin for 5g and beyond. *IEEE Communications Magazine*, 59(2):10–15, 2021.

M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 809–816, 2011.

M. Nikolaou. Model predictive controllers: A critical synthesis of theory and industrial needs. *Advances in Chemical Engineering*, 26:131–204, 2001.

N. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor. Industry-scale knowledge graphs: lessons and challenges. *Communications of the ACM*, 62(8):36–43, 2019.

H. Oh, J. H. Jung, B. C. Jeon, and B. D. Youn. Scalable and unsupervised feature engineering using vibration-imaging and deep learning for rotor system diagnosis. *IEEE Transactions on Industrial Electronics*, 65(4):3539–3549, 2017.

S. E. Otto and C. W. Rowley. Linearly recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1):558–593, 2019.

G. B. Ozturk. Digital twin research in the aeco-fm industry. *Journal of Building Engineering*, 40:102730, 2021.

È. Pairet, P. Ardón, X. Liu, J. Lopes, H. Hastie, and K. S. Lohan. A digital twin for human-robot interaction. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 372–372. IEEE, 2019.

Y. Pan and L. Zhang. A bim-data mining integrated digital twin framework for advanced project management. *Automation in Construction*, 124:103564, 2021.

J. Park, J. M. Ha, H. Oh, B. D. Youn, J.-H. Choi, and N. H. Kim. Model-based fault diagnosis of a planetary gear: A novel approach using transmission error. *IEEE Transactions on Reliability*, 65(4):1830–1841, 2016.

B. Peeters and G. De Roeck. Reference-based stochastic subspace identification for output-only modal analysis. *Mechanical Systems and Signal Processing*, 13(6):855–878, 1999.

B. Peherstorfer, K. Willcox, and M. Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, 60(3):550–591, 2018.

F.-Q. Pei, Y.-F. Tong, M.-H. Yuan, K. Ding, and X.-H. Chen. The digital twin of the quality monitoring and control in the series solar cell production line. *Journal of Manufacturing Systems*, 59:127–137, 2021.

W. Peng, Y.-F. Li, Y.-J. Yang, S.-P. Zhu, and H.-Z. Huang. Bivariate analysis of incomplete degradation observations based on inverse gaussian processes and copulas. *IEEE Transactions on Reliability*, 65(2):624–639, 2016.

Y. Peng, S. Zhao, and H. Wang. A digital twin based estimation method for health indicators of dc–dc converters. *IEEE Transactions on Power Electronics*, 36(2):2105–2118, 2020.

Z. K. Peng and F. Chu. Application of the wavelet transform in machine condition monitoring and fault diagnostics: a review with bibliography. *Mechanical Systems and Signal Processing*, 18 (2):199–221, 2004.

H. T. Pham, B.-S. Yang, T. T. Nguyen, et al. Machine performance degradation assessment and remaining useful life prediction using proportional hazard model and support vector machine. *Mechanical Systems and Signal Processing*, 32:320–330, 2012.

L. Piroddi and W. Spinelli. An identification algorithm for polynomial narx models based on simulation error minimization. *International Journal of Control*, 76(17):1767–1781, 2003.

G. L. Plett. Extended kalman filtering for battery management systems of lipb-based hev battery packs: Part 3. state and parameter estimation. *Journal of Power sources*, 134(2):277–292, 2004.

E. Priyanka, S. Thangavel, X.-Z. Gao, and N. Sivakumar. Digital twin for oil pipeline risk estimation using prognostic and machine learning techniques. *Journal of Industrial Information Integration*, 26:100272, 2022.

F. Psarommatis. A generic methodology and a digital twin for zero defect manufacturing (zdm) performance mapping towards design for zdm. *Journal of Manufacturing Systems*, 59:507–521, 2021.

G. V. Puskorius and L. A. Feldkamp. Neurocontrol of nonlinear dynamical systems with kalman filter trained recurrent networks. *IEEE Transactions on Neural Networks*, 5(2):279–297, 1994.

Q. Qi and F. Tao. Digital twin and big data towards smart manufacturing and industry 4.0:

360 degree comparison. *IEEE Access*, 6:3585–3593, 2018.

Q. Qi, F. Tao, Y. Zuo, and D. Zhao. Digital twin service towards smart manufacturing. *Procedia CIRP*, 72:237–242, 2018.

Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, and A. Nee. Enabling technologies and tools for digital twin. *Journal of Manufacturing Systems*, 58:3–21, 2021.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019a.

M. Raissi, Z. Wang, M. S. Triantafyllou, and G. E. Karniadakis. Deep learning of vortex-induced vibrations. *Journal of Fluid Mechanics*, 861: 119–137, 2019b.

V. Ramadesigan, K. Chen, N. A. Burns, V. Boovaragavan, R. D. Braatz, and V. R. Subramanian. Parameter estimation and capacity fade analysis of lithium-ion batteries using reformulated models. *Journal of the Electrochemical Society*, 158(9):A1048, 2011.

A. L. Ramos, J. V. Ferreira, and J. Barceló. Model-based systems engineering: An emerging approach for modern systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(1):101–111, 2011.

R. B. Randall. *Vibration-based condition monitoring: industrial, automotive and aerospace applications*. John Wiley & Sons, 2021.

R. B. Randall and J. Antoni. Rolling element bearing diagnostics—a tutorial. *Mechanical Systems and Signal Processing*, 25(2):485–520, 2011.

A. Rasheed, O. San, and T. Kvamsdal. Digital twin: Values, challenges and enablers from a modeling perspective. *IEEE Access*, 8:21980–22012, 2020.

A. Rassõlkin, T. Vaimann, A. Kallaste, and V. Kuts. Digital twin for propulsion drive of autonomous electric vehicle. In *2019 IEEE 60th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, pages 1–4. IEEE, 2019.

R. Revetria, F. Tonelli, L. Damiani, M. Demartini, F. Bisio, and N. Peruzzo. A real-time mechanical structures monitoring system based on digital twin, iot and augmented reality. In *2019 Spring Simulation Conference (SpringSim)*, pages 1–10. IEEE, 2019.

S. M. Rezvanizaniani, Z. Liu, Y. Chen, and J. Lee. Review and recent advances in battery health monitoring and prognostics technologies for electric vehicle (ev) safety and mobility. *Journal of Power Sources*, 256:110–124, 2014.

R. R. Richardson, M. A. Osborne, and D. A. Howey. Gaussian process regression for forecasting battery state of health. *Journal of Power Sources*, 357:209–219, 2017.

I. Ricondo, A. Porto, and M. Ugarte. A digital twin framework for the simulation and optimization of production systems. *Procedia CIRP*, 104:762–767, 2021.

T. Ritto and F. Rochinha. Digital twin, physics-based model, and machine learning applied to damage detection in structures. *Mechanical Systems and Signal Processing*, 155:107614, 2021.

J. L. Rojo-Álvarez, M. Martínez-Ramón, M. de Prado-Cumplido, A. Artés-Rodríguez, and A. R. Figueiras-Vidal. Support vector method for robust arma system identification. *IEEE Transactions on Signal Processing*, 52 (1):155–164, 2004.

K. Rombach, G. Michau, and O. Fink. Controlled generation of unseen faults for partial and openset&partial domain adaptation. *arXiv preprint arXiv:2204.14068*, 2022.

M. Russell and P. Wang. Physics-informed deep learning for signal compression and reconstruction of big data in industrial condition monitoring. *Mechanical Systems and Signal Processing*, 168:108709, 2022.

M. Sadoughi and C. Hu. Physics-based convolutional neural network for fault diagnosis of rolling element bearings. *IEEE Sensors Journal*, 19(11):4181–4192, 2019.

P. D. Samuel and D. J. Pines. A review of vibration-based techniques for helicopter transmission diagnostics. *Journal of sound and vibration*, 282(1-2):475–508, 2005.

S. Sankararaman, Y. Ling, and S. Mahadevan. Uncertainty quantification and model validation of fatigue crack growth prediction. *Engineering Fracture Mechanics*, 78(7):1487–1504, 2011.

S. Särkkä. *Bayesian filtering and smoothing*. Number 3. Cambridge university press, 2013.

B. Schleich, N. Anwer, L. Mathieu, and S. Wartzack. Shaping the digital twin for design and production engineering. *CIRP Annals*, 66 (1):141–144, 2017.

S. Schoenmackers, J. Davis, O. Etzioni, and D. S. Weld. Learning first-order horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1088–1098, 2010.

G. Schrotter and C. Hürzeler. The digital twin of the city of zurich for urban planning. *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88(1):99–112, 2020.

M. Schwenzer, M. Ay, T. Bergs, and D. Abel. Review on model predictive control: an engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117: 1327–1349, 2021.

M. Scutari. Learning bayesian networks with the bnlearn r package. *arXiv preprint arXiv:0908.3817*, 2009.

B. R. Seshadri and T. Krishnamurthy. Structural health management of damaged aircraft structures using digital twin concept. In *25th aiaa/ahs adaptive structures conference*, page 1675, 2017.

S. Shao, P. Wang, and R. Yan. Generative adversarial networks for data augmentation in machine fault diagnosis. *Computers in Industry*, 106:85–93, 2019.

S. Shen, M. Sadoughi, M. Li, Z. Wang, and C. Hu. Deep convolutional neural networks with ensemble learning and transfer learning for capacity estimation of lithium-ion batteries. *Applied Energy*, 260:114296, 2020.

S. Shen, H. Lu, M. Sadoughi, C. Hu, V. Nemani, A. Thelen, K. Webster, M. Darr, J. Sidon, and S. Kenny. A physics-informed deep learning approach for bearing fault detection. *Engineering Applications of Artificial Intelligence*, 103: 104295, 2021.

C.-S. Shim, N.-S. Dang, S. Lon, and C.-H. Jeon. Development of a bridge maintenance system for prestressed concrete bridges using 3d digital twin model. *Structure and Infrastructure Engineering*, 15(10):1319–1332, 2019.

D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, pages 387–395. PMLR, 2014.

D. Simon. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

G. Sirca Jr and H. Adeli. System identification in structural engineering. *Scientia Iranica*, 19(6): 1355–1364, 2012.

A. Smith, J. C. Burns, X. Zhao, D. Xiong, and J. Dahn. A high precision coulometry study of the sei growth in li/graphite cells. *Journal of The Electrochemical Society*, 158(5):A447, 2011.

R. Söderberg, K. Wärmefjord, J. S. Carlson, and L. Lindkvist. Toward a digital twin for real-time geometry assurance in individualized production. *CIRP Annals*, 66(1):137–140, 2017.

C. Steinmetz, A. Rettberg, F. G. C. Ribeiro, G. Schroeder, and C. E. Pereira. Internet of things ontology for digital twin in cyber physical systems. In *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC)*, pages 154–159. IEEE, 2018.

M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe. Direct importance estimation for covariate shift adaptation. *Annals of the Institute of Statistical Mathematics*, 60(4):699–746, 2008.

W. Sun, S. Lei, L. Wang, Z. Liu, and Y. Zhang. Adaptive federated learning and digital twin for industrial internet of things. *IEEE Transactions on Industrial Informatics*, 17(8):5605–5614, 2020a.

X. Sun, J. Bao, J. Li, Y. Zhang, S. Liu, and B. Zhou. A digital twin-driven approach for the assembly-commissioning of high precision products. *Robotics and Computer-Integrated Manufacturing*, 61:101839, 2020b.

A. Svensson. Partilce-filter-video. *https://github.com/uu-sml/partilce-filter-video*, Last Accessed on April 9, 2022, 2022a.

A. Svensson. Particle filter explained without equations. *https://www.youtube.com/watch?v=aUkBa1zMKv4.*, Last Accessed on April 9, 2022, 2022b.

A. Swief, A. El-Zawawi, and M. El-Habrouk. A survey of model predictive control development in automotive industries. 2019.

S. K. Tadeja, P. Seshadri, and P. O. Kristensson. Aerovr: An immersive visualisation system for aerospace design and digital twinning in virtual reality. *The Aeronautical Journal*, 124(1280): 1615–1635, 2020.

M. Tang, Y. Liu, and L. J. Durlofsky. A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems. *Journal of Computational Physics*, 413:109456, 2020.

F. Tao and M. Zhang. Digital twin shop-floor: a new shop-floor paradigm towards smart manufacturing. *IEEE Access*, 5:20418–20427, 2017.

F. Tao, H. Zhang, A. Liu, and A. Y. Nee. Digital twin in industry: State-of-the-art. *IEEE Transactions on Industrial Informatics*, 15(4): 2405–2415, 2018.

E. Taskesen. Bnlearn - library for bayesian network learning and inference. https://pypi.org/project/bnlearn/. Last Accessed: 2022-04-24.

T. Tezduyar and Y. Osawa. Fluid–structure interactions of a parachute crossing the far wake of an aircraft. *Computer Methods in Applied Mechanics and Engineering*, 191(6-7):717–726, 2001.

A. Thelen, Y. H. Lui, S. Shen, S. Laflamme, S. Hu, and C. Hu. Physics-informed machine learning for degradation diagnostics of lithium-ion batteries. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 85383, page V03AT03A041. American Society of Mechanical Engineers, 2021.

A. Thelen, Y. H. Lui, S. Shen, S. Laflamme, S. Hu, and C. Hu. Integrating physics-based modeling and machine learning for degradation diagnostics of lithium-ion batteries. *Energy Storage Materials*, 2022a.

A. Thelen, X. Zhang, O. Fink, Y. Lu, S. Ghosh, B. D. Youn, M. D. Todd, S. Mahadevan, C. Hu, and Z. Hu. A comprehensive review of digital twin - part 2: Roles of uncertainty quantification and optimization, perspectives, and a case study on a battery digital twin. *Structural and Multidisciplinary Optimization*, 2022b.

J. Tian, R. Xiong, W. Shen, and F. Sun. Electrode ageing estimation and open circuit voltage reconstruction for lithium ion batteries. *Energy Storage Materials*, 37:283–295, 2021.

Y. Tian, M. A. Chao, C. Kulkarni, K. Goebel, and O. Fink. Real-time model calibration with deep reinforcement learning. *Mechanical Systems and Signal Processing*, 165:108284, 2022a.

Y. Tian, M. Han, C. Kulkarni, and O. Fink. A prescriptive dirichlet power allocation policy with deep reinforcement learning. *Reliability Engineering & System Safety*, 224:108529, 2022b.

Z. Tian. An artificial neural network method for remaining useful life prediction of equipment subject to condition monitoring. *Journal of intelligent Manufacturing*, 23(2):227–237, 2012.

Z. Tian, T. Jin, B. Wu, and F. Ding. Condition based maintenance optimization for wind power generation systems under continuous monitoring. *Renewable Energy*, 36(5):1502–1509, 2011.

S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM workshop on Artificial Intelligence and Security*, pages 1–11, 2019.

A. Tsymbal. The problem of concept drift: definitions and related work. *Computer Science Department, Trinity College Dublin*, 106(2):58, 2004.

E. J. Tuegel, A. R. Ingraffea, T. G. Eason, and S. M. Spottswood. Reengineering aircraft structural life prediction using a digital twin. *International Journal of Aerospace Engineering*, 2011, 2011.

E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

T. H.-J. Uhlemann, C. Lehmann, and R. Steinhilper. The digital twin: Realizing the cyber-physical production system for industry 4.0. *Procedia CIRP*, 61:335–340, 2017.

P. Van Hentenryck. Machine learning for optimal power flows. *Tutorials in Operations Research: Emerging Optimization Methods and Modeling Techniques with Applications*, pages 62–82, 2021.

E. VanDerHorn and S. Mahadevan. Digital twin: Generalization, characterization and implementation. *Decision Support Systems*, 145:113524, 2021.

A. Vatankhah Barenji, X. Liu, H. Guo, and Z. Li. A digital twin-driven approach towards smart manufacturing: Reduced energy consumption for a robotic cell. *International Journal of Computer Integrated Manufacturing*, 34(7-8): 844–859, 2021.

M. Vathoopan, M. Johny, A. Zoitl, and A. Knoll. Modular fault ascription and corrective maintenance using a digital twin. *IFAC-PapersOnLine*, 51(11):1041–1046, 2018.

M. A. Vega, Z. Hu, and M. D. Todd. Optimal maintenance decisions for deteriorating quoin blocks in miter gates subject to uncertainty in the condition rating protocol. *Reliability Engineering & System Safety*, 204:107147, 2020.

M. A. Vega, Z. Hu, T. B. Fillmore, M. D. Smith, and M. D. Todd. A novel framework for integration of abstracted inspection data and structural health monitoring for damage prognosis of miter gates. *Reliability Engineering & System Safety*, 211:107561, 2021.

M. A. Vega, Z. Hu, Y. Yang, M. Chadha, and M. D. Todd. Diagnosis, prognosis, and maintenance decision making for civil infrastructure: Bayesian data analytics and machine learning. In *Structural Health Monitoring Based on Data Science Techniques*, pages 45–73. Springer, 2022.

W. Waag, C. Fleischer, and D. U. Sauer. Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles. *Journal of Power Sources*, 258:321–339, 2014.

D. Wagg, K. Worden, R. Barthorpe, and P. Gardner. Digital twins: state-of-the-art and future directions for modeling and simulation in engineering dynamics applications. *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg*, 6(3), 2020.

P. Wagner, X. Wu, and M. F. Huber. Kalman bayesian neural networks for closed-form online learning. *arXiv preprint arXiv:2110.00944*, 2021.

E. A. Wan and A. T. Nelson. Dual extended kalman filter methods. *Kalman filtering and neural networks*, 123, 2001.

D. Wang, Q. Miao, and M. Pecht. Prognostics of lithium-ion batteries based on relevance vectors and a conditional three-parameter capacity degradation model. *Journal of Power Sources*, 239:253–264, 2013.

J. Wang, L. Ye, R. X. Gao, C. Li, and L. Zhang. Digital twin for rotating machinery fault diagnosis in smart manufacturing. *International Journal of Production Research*, 57(12):3920–3934, 2019.

M. Wang, S. Feng, A. Incecik, G. Królczyk, and Z. Li. Structural fatigue life prediction considering model uncertainties through a novel digital twin-driven approach. *Computer Methods in Applied Mechanics and Engineering*, 391: 114512, 2022a.

Q. Wang, Z. Mao, B. Wang, and L. Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

Q. Wang, W. Jiao, P. Wang, and Y. Zhang. Digital twin for human-robot interactive welding and welder behavior analysis. *IEEE/CAA Journal of Automatica Sinica*, 8(2):334–343, 2020a.

Q. Wang, W. Jiao, and Y. Zhang. Deep learning-empowered digital twin for visualized weld joint growth monitoring and penetration control. *Journal of Manufacturing Systems*, 57:429–439, 2020b.

Q. Wang, G. Michau, and O. Fink. Missing-class-robust domain adaptation by unilateral alignment. *IEEE Transactions on Industrial Electronics*, 68:663–671, 2020c.

Q. Wang, C. Taal, and O. Fink. Integrating expert knowledge with domain adaptation for unsupervised fault diagnosis. *IEEE Transactions on Instrumentation and Measurement*, 2021a.

Y. Wang, R. Xu, C. Zhou, X. Kang, and Z. Chen. Digital twin and cloud-side-end collaboration for intelligent battery management system. *Journal of Manufacturing Systems*, 62:124–134, 2022b.

Z. Wang, Q. Liu, H. Chen, and X. Chu. A deformable cnn-dlstm based transfer learning method for fault diagnosis of rolling bearing under multiple working conditions. *International Journal of Production Research*, 59(16): 4811–4825, 2021b.

R. Ward, R. Choudhary, A. Gregory, M. Jans-Singh, and M. Girolami. Continuous calibration of a digital twin: Comparison of particle filter and bayesian calibration approaches. *Data-Centric Engineering*, 2, 2021.

K. Wärmefjord, R. Söderberg, L. Lindkvist, B. Lindau, and J. S. Carlson. Inspection data to support a digital twin for geometry assurance. In *ASME International Mechanical Engineering Congress and Exposition*, volume 58356, page V002T02A101. American Society of Mechanical Engineers, 2017.

T. Weber, A. Corotan, B. Hutchinson, B. Kravitz, and R. Link. Deep learning for creating surrogate models of precipitation in earth system models. *Atmospheric Chemistry and Physics*, 20(4):2303–2317, 2020.

Z. Wei, Z. Quan, J. Wu, Y. Li, J. Pou, and H. Zhong. Deep deterministic policy gradient-drl enabled multiphysics-constrained fast charging of lithium-ion battery. *IEEE Transactions on Industrial Electronics*, 69(3):2588–2598, 2021.

T. Weilkiens. *Systems engineering with SysML/UML: modeling, analysis, design*. Elsevier, 2011.

T. A. Wenzel, K. Burnham, M. Blundell, and R. Williams. Dual extended kalman filter for vehicle state and parameter estimation. *Vehicle System Dynamics*, 44(2):153–171, 2006.

G. White, A. Zink, L. Codecá, and S. Clarke. A digital twin smart city for citizen feedback. *Cities*, 110:103064, 2021.

D. V. Widder. *The heat equation*, volume 67. Academic Press, 1976.

J. Willard, X. Jia, S. Xu, M. Steinbach, and V. Kumar. Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919*, 1(1):1–34, 2020.

W. C. Wong, J. Li, and X. Wang. Recurrent neural network-based model predictive control for continuous pharmaceutical manufacturing, 2018.

K. Worden, W. Becker, T. Rogers, and E. Cross. On the confidence bounds of gaussian process narx models and their higher-order frequency response functions. *Mechanical Systems and Signal Processing*, 104:188–223, 2018.

L. Wright and S. Davidson. How to tell the difference between a model and a digital twin. *Advanced Modeling and Simulation in Engineering Sciences*, 7(1):1–13, 2020.

R.-T. Wu and M. R. Jahanshahi. Deep convolutional neural network for structural dynamic response estimation and system identification. *Journal of Engineering Mechanics*, 145(1):04018125, 2019.

Z. Wu, D. Rincon, and P. D. Christofides. Real-time adaptive machine-learning-based predictive control of nonlinear processes. *Industrial & Engineering Chemistry Research*, 59(6):2275–2290, 2019a.

Z. Wu, S. Yu, X. Zhu, Y. Ji, and M. Pecht. A weighted deep domain adaptation method for industrial fault prognostics according to prior distribution of complex working conditions. *IEEE Access*, 7:139802–139814, 2019b.

K. Xia, C. Sacco, M. Kirkpatrick, C. Saidy, L. Nguyen, A. Kircaliali, and R. Harik. A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence. *Journal of Manufacturing Systems*, 58:210–230, 2021a.

M. Xia, H. Shao, D. Williams, S. Lu, L. Shu, and C. W. de Silva. Intelligent fault diagnosis of machinery using digital twin-assisted deep transfer learning. *Reliability Engineering & System Safety*, 215:107938, 2021b.

R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Y. Xie, K. Lian, Q. Liu, C. Zhang, and H. Liu. Digital twin for cutting tool: Modeling, application and service strategy. *Journal of Manufacturing Systems*, 58:305–312, 2021.

Y. Xu, Y. Sun, X. Liu, and Y. Zheng. A digital-twin-assisted fault diagnosis using deep transfer learning. *IEEE Access*, 7:19990–19999, 2019.

L. Xuemei, D. Lixing, D. Yuyuan, and L. Lanlan. Hybrid support vector machine and arima model in building cooling prediction. In *2010 International Symposium on Computer, Communication, Control and Automation (3CA)*, volume 1, pages 533–536. IEEE, 2010.

H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2272–2281, 2017.

T. Yan, Y. Fu, M. Lu, Z. Li, C. Shen, and D. Wang. Integration of a novel knowledge-guided loss function with an architecturally explainable network for machine degradation modeling. *IEEE Transactions on Instrumentation and Measurement*, 2022.

Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):Article 19, 2019.

X. Yang, G. Tartakovsky, and A. Tartakovsky. Physics-informed kriging: A physics-informed gaussian process regression method for data-model convergence. *arXiv preprint arXiv:1809.03461*, 2018.

Z. Yang, W. W. Cohen, and R. Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016.

H. Yao, Y. Gao, and Y. Liu. Fea-net: A physics-guided data-driven model for efficient mechanical response prediction. *Computer Methods in Applied Mechanics and Engineering*, 363: 112892, 2020.

Y. Ye, Q. Yang, F. Yang, Y. Huo, and S. Meng. Digital twin for the structural health management of reusable spacecraft: A case study. *Engineering Fracture Mechanics*, 234:107076, 2020.

Z.-S. Ye and M. Xie. Stochastic modelling and analysis of degradation for highly reliable products. *Applied Stochastic Models in Business and Industry*, 31(1):16–32, 2015.

Y. Yi, Y. Yan, X. Liu, Z. Ni, J. Feng, and J. Liu. Digital twin-based smart assembly process design and application framework for complex products and its case study. *Journal of Manufacturing Systems*, 58:94–107, 2021.

X. Yin, Y. Zhu, and J. Hu. A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys (CSUR)*, 54(6):1–36, 2021.

J. Yu, P. Liu, and Z. Li. Hybrid modelling and digital twin development of a steam turbine control stage for online performance monitoring. *Renewable and Sustainable Energy Reviews*, 133:110077, 2020.

J. Yu, Y. Song, D. Tang, and J. Dai. A digital twin approach based on nonparametric bayesian network for complex system health monitoring. *Journal of Manufacturing Systems*, 58:293–304, 2021.

M. Yuan, Y. Wu, and L. Lin. Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network. In *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*, pages 135–140. IEEE, 2016.

Y. A. Yucesan and F. A. Viana. A physics-informed neural network for wind turbine main bearing fatigue. *International Journal of Prognostics and Health Management*, 11(1), 2020.

Y. A. Yucesan and F. A. Viana. A hybrid physics-informed neural network for main bearing fatigue prognosis under grease quality variation. *Mechanical Systems and Signal Processing*, 171:108875, 2022.

S. Zambal, C. Eitzinger, M. Clarke, J. Klintworth, and P.-Y. Mechin. A digital twin for composite parts manufacturing: Effects of defects analysis based on manufacturing data. In *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, pages 803–808. IEEE, 2018.

A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36(4):1307–1319, 2020.

A. Zhang, H. Wang, S. Li, Y. Cui, Z. Liu, G. Yang, and J. Hu. Transfer learning with deep recurrent neural networks for remaining useful life estimation. *Applied Sciences*, 8(12):2416, 2018.

H. Zhang, X. Wang, C. Zhang, and X. Cai. Robust identification of non-linear dynamic systems using support vector machine. *IEE Proceedings-Science, Measurement and Technology*, 153(3): 125–129, 2006.

H. Zhang, Q. Qi, and F. Tao. A multi-scale modeling method for digital twin shop-floor. *Journal of Manufacturing Systems*, 62:417–428, 2022a.

K. Zhang, T. Qu, D. Zhou, H. Jiang, Y. Lin, P. Li, H. Guo, Y. Liu, C. Li, and G. Q. Huang. Digital twin-based opti-state control method for a synchronized production operation system. *Robotics and Computer-Integrated Manufacturing*, 63:101892, 2020a.

L. Zhang, L. Zhou, and B. K. Horn. Building a right digital twin with model engineering. *Journal of Manufacturing Systems*, 59:151–164, 2021a.

M. Zhang, F. Tao, and A. Nee. Digital twin enhanced dynamic job-shop scheduling. *Journal of Manufacturing Systems*, 58:146–156, 2021b.

P. Zhang and Z.-Y. Yin. A novel deep learning-based modelling strategy from image of particles to mechanical properties for granular materials with cnn and bilstm. *Computer Methods in Applied Mechanics and Engineering*, 382: 113858, 2021.

S. Zhang, S. Zhang, B. Wang, and T. G. Habetler. Deep learning algorithms for bearing fault diagnostics—a comprehensive review. *IEEE Access*, 8:29857–29881, 2020b.

X. Zhang and S. Mahadevan. Bayesian neural networks for flight trajectory prediction and safety assessment. *Decision Support Systems*, 131: 113246, 2020.

X. Zhang, F. T. Chan, and S. Mahadevan. Explainable machine learning in image classification models: An uncertainty quantification perspective. *Knowledge-Based Systems*, 243: 108418, 2022b.

X. Zhang, F. T. Chan, C. Yan, and I. Bose. Towards risk-aware artificial intelligence and machine learning systems: An overview. *Decision Support Systems*, page 113800, 2022c.

Y. Zhang, S. Wang, G. Sun, and J. Mao. Aerodynamic surrogate model based on deep long short-term memory network: An application on high-lift device control. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, page 09544100211027023, 2021c.

Z. Zhang, Y. Wang, and K. Wang. Fault diagnosis and prognosis using wavelet packet decomposition, fourier transform and artificial neural network. *Journal of Intelligent Manufacturing*, 24(6):1213–1227, 2013.

R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao. Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, 115:213–237, 2019.

Y. Zhao, C. Jiang, M. A. Vega, M. D. Todd, and Z. Hu. Surrogate modeling of nonlinear dynamic systems: A comparative study. *Journal of Computing and Information Science in Engineering*, pages 1–48, 2022.

Y. Zheng, S. Yang, and H. Cheng. An application framework of digital twin and its case study. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1141–1153, 2019.

T. Zhou, X. Zhang, E. L. Droguett, and A. Mosleh. A generic physics-informed neural network-based framework for reliability assessment of multi-state systems. *Reliability Engineering & Systems Safety*, 2022a.

X. Zhou, S. He, L. Dong, and S. N. Atluri. Real-time prediction of probabilistic crack growth with a helicopter component digital twin. *AIAA Journal*, pages 1–13, 2022b.

Y. Zhou, T. Xing, Y. Song, Y. Li, X. Zhu, G. Li, and S. Ding. Digital-twin-driven geometric optimization of centrifugal impeller with free-form blades for five-axis flank milling. *Journal of Manufacturing Systems*, 58:22–35, 2021.

L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019a.

Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, and P. Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394:56–81, 2019b.

Z. Zhu, X. Xi, X. Xu, and Y. Cai. Digital twin-driven machining process for thin-walled part manufacturing. *Journal of Manufacturing Systems*, 59:453–466, 2021.

C. Zhuang, J. Liu, and H. Xiong. Digital twin-based smart production management and control framework for the complex product assembly shop-floor. *The International Journal of Advanced Manufacturing Technology*, 96(1): 1149–1163, 2018.

T. Zohdi. A machine-learning framework for rapid adaptive digital-twin based fire-propagation simulation in complex environments. *Computer Methods in Applied Mechanics and Engineering*, 363:112907, 2020.