Journal of Intelligent Manufacturing (2022) 33:2027–2043
https://doi.org/10.1007/s10845-022-01952-x

Understanding unforeseen production downtimes in manufacturing processes using log data-driven causal reasoning

Christopher Hagedorn[1] · **Johannes Huegle**[1] · **Rainer Schlosser**[1]

Received: 6 September 2021 / Accepted: 12 April 2022 / Published online: 23 May 2022

Abstract
In discrete manufacturing, the knowledge about causal relationships makes it possible to avoid unforeseen production downtimes by identifying their root causes. Learning causal structures from real-world settings remains challenging due to high-dimensional data, a mix of discrete and continuous variables, and requirements for preprocessing log data under the causal perspective. In our work, we address these challenges proposing a process for causal reasoning based on raw machine log data from production monitoring. Within this process, we define a set of transformation rules to extract independent and identically distributed observations. Further, we incorporate a variable selection step to handle high-dimensionality and a discretization step to include continuous variables. We enrich a commonly used causal structure learning algorithm with domain-related orientation rules, which provides a basis for causal reasoning. We demonstrate the process on a real-world dataset from a globally operating precision mechanical engineering company. The dataset contains over 40 million log data entries from production monitoring of a single machine. In this context, we determine the causal structures embedded in operational processes. Further, we examine causal effects to support machine operators in avoiding unforeseen production stops, i.e., by detaining machine operators from drawing false conclusions on impacting factors of unforeseen production stops based on experience.

Keywords Causal structure learning · Log data · Causal inference · Manufacturing industry

Introduction

Production downtime is one of the most significant contributors to production inefficiency (Liu et al., 2012), resulting in lost profit. While planned production downtime occurs, for example, for scheduled maintenance based on regular schedules or predictive models (Boudjelida, 2019; Khatab, 2018; Liu et al., 2019; Sipos et al., 2014), unforeseen production stops are a result of failures in the production process, e.g., misconfiguration of a machine, intervention of a worker, or defective raw material. In the case of an unforeseen production stop, direct action from production workers is required to resolve the issue promptly and limit the financial loss (Mobley, 2002). Therefore, knowing the reason for the production stop and understanding the root cause supports resolving the issue effectively. Furthermore, the corresponding knowledge about the causal structures supports the machine operator to take useful precautionary measures to avoid future production stops.

Modern discrete manufacturing companies aim for increased product quality, diversified products, reduced cost, and lower manufacturing time, while at the same time being faced by shortened product life-cycles and global competition (Chen & Huang, 2006; Liang et al., 2004; Wuest et al., 2014). These goals are reflected in production machines that provide hundreds of configuration parameters. The introduced complexity in the machine's operation becomes challenging for the human operator to handle. The complexity rises further as, driven by the Internet of Things (IoT), an increasing amount of data is generated

Christopher Hagedorn and Johannes Huegle have contributed equally to this work.

✉ Christopher Hagedorn
christopher.hagedorn@hpi.de

Johannes Huegle
johannes.huegle@hpi.de

Rainer Schlosser
rainer.schlosser@hpi.de

[1] Chair for Enterprise Platform and Integration Concepts, Hasso Plattner Institute, University of Potsdam, August-Bebel Str. 88, 14482 Potsdam, Brandenburg, Germany

during production (Marazopoulou et al., 2016; Qin et al., 2020), e.g., coming from shop floor systems, production machinery, robots, or sensors. This information is commonly stored while monitoring the production process and the product quality to detect defects. Harnessing this log data vault beyond monitoring opens the opportunity for a data-driven examination of predictive maintenance or automatic Root Cause Analysis (RCA), e.g., for increasing production efficiency, reducing defects, or decreasing unforeseen production downtime (Davis et al., 2015; Gutschi et al., 2019; Li et al., 2020; Nikula et al., 2019; Rodríguez et al., 2019; Sipos et al., 2014; Sun et al., 2021; Wang et al., 2017;Wuest et al., 2016). In this context, random forests are used to derive models for predicting general machine breakdown (Gutschi et al., 2019; Wang et al., 2017), or they are used to select possible causes of faults within a manufacturing process (Chien & Chuang, 2014). Other approaches, based on neural networks (Du et al., 2012; Nikula et al., 2019) or deep belief networks (Li et al., 2020), focus on predicting specific mechanical issues within machines as an indicator for required maintenance. Clustering techniques are a way to diagnose faults in mechanical systems to retain productivity (Rodríguez et al., 2019) or to find fault-generating combinations of machines (Rokach & Hutter, 2012).

Usually most methods used for predictive maintenance and automatic RCA rely on associational patterns within the observational data of the production process (e Oliveira et al., 2022). In recent years, the emergence of methods for causal reasoning enable a data-driven examination of causal structures and causal effects beyond associational patterns within observational data (Hernán & Robins, 2020; Pearl, 2009b; Peters et al., 2017; Spirtes et al., 2000). In this context, a causal graphical model depicts the respective causal structures and is the basis for causal effect estimation. Understanding the causal structures in complex manufacturing settings supports to find root causes of faults, which in practice allows machine operators to address unforeseen production stops effectively, e.g., see Kühnert and Beyerer (2014), Marazopoulou et al. (2016), Ye (2017), or Huegle et al. (2020). Moreover, the causal effect estimation based on learned causal structures points the machine operator to relevant adjustments or repairs within the production process, e.g., see Sun et al. (2021) or e Oliveira et al. (2021). For a detailed overview on recent advances and methods used for automatic RCA we refer to e Oliveira (2022). In their work, e Oliveira (2022) point out the need to consider methods that allow causal conclusions and do not rely on associational patterns within the observational data. In this context, they mention the challenges of a data-driven examination of causal structures or causal effects and state a research gap that focuses on the application of these methods on RCA in manufacturing processes.

In our work, we close this research gap and apply causal structure learning and causal effect estimation on log data within a real-world scenario and data from a globally operating precision mechanical engineering company. The company produces large manufacturing machines and supports the customers who operate the machinery. We focused our running example on a single machine for simplicity, yet we discovered the same patterns based on the data from similar machines of the precision mechanical engineering company.

Challenges in log data-driven causal structure learning

Within the scenario at hand, we find three relevant challenges to log data-driven causal structure learning that are common in practice (Malinsky & Danks, 2018). A machine logs its configuration parameters, internal state based on sensor readings, and error messages during production. Thus, the machine contains millions of entries with several thousand different types, resulting in high-dimensional data (Challenge I). High-dimensional data leads to long execution times, hindering the application of causal structure learning in practice (Le et al., 2019). Further, it increases the potential for statistical error (Maathuis et al., 2018). The data is recorded at different time intervals and eventually stored in a semi-structured log format. The semi-structured machine data needs to be preprocessed (Wuest et al., 2016) before the application of causal structure learning to extract the independent and identically distributed (i.i.d) observational data (Spirtes & Zhang, 2016) (Challenge II). Further, the machine log data contains a mixture of continuous variables, such as sensor measurements, and discrete variables, such as configuration parameters or error messages. While recently developed methods for causal structure learning work on mixed data with continuous and discrete variables (Huegle, 2021), they are often not considered in practice due to high computational requirements (Challenge III).

Contribution

We propose a process to learn causal structures from machine log data, addressing the challenges mentioned above. The process consists of three steps, a preprocessing procedure, a procedure to learn the causal structures, and optional causal effect estimation. Within the preprocessing procedure, we define a set of transformation rules as a preprocessing step to obtain independent and identically distributed (i.i.d.) observations (cf. Challenge II). We integrate additional processing steps into the causal structure learning procedure to handle the mixed and high-dimensional data incorporating domain expertise (cf. Challenges I and III). Despite these additions, the causal structure learning procedure follows common constraint-based methods that leverage conditional indepen-

dence information for learning the structure. Further, we define rules for edge orientation based on process-specific and engineering-specific knowledge. Our contributions can be summarized as follows:

1. Considering a real-world production use case, we show how the causes of unforeseen production downtimes can be revealed using causal structure learning.
2. We demonstrate causal effect estimation's applicability and effectiveness in an experimental regime using the learned causal structures from log data.
3. The concepts used within the proposed process are domain-independent and general enough to apply to other manufacturing industries.

The remainder of the paper is structured as follows. In Sect. 2, we discuss related work. In Sect. 3, we provide the theoretical background on causal reasoning. The real-world scenario of unforeseen production stops and the corresponding machine log data utilized in our work are described in Sect. 4. Section 5 explains the transformation rules applied to the machine log data and our proposed algorithmic approach to learning causal structures based on the extracted data. We demonstrate each process step for our real-world example, including applying methods of causal inference based on the learned causal structures. We conclude our work in Sect. 6.

Causal reasoning in the manufacturing domain

Several studies have considered the application of causal structure learning for causal reasoning in specific use cases in the manufacturing domain (Huegle et al., 2020; Kühnert & Beyerer, 2014; Li & Shi, 2007; Marazopoulou et al., 2016; Ye, 2017). Each work focuses on improvements concerning the particular use case, input data, or application. Some work incorporates preprocessing of the raw manufacturing data and utilizes domain-specific background knowledge. For an overview comparing the existing literature, see Table 1. To the best of our knowledge, there is no overall approach for causal reasoning in the manufacturing domain, which starts with raw log data from production monitoring and includes the application of causal inference. For general best practices for causal discovery on real-world data, we refer to Malinsky & Danks (2018).

Li and Shi (2007) focus on the identification of causal structures in a rolling process. They use product quality and process data to derive causal relationships to facilitate process control. Their work proposes adaptions to the PC algorithm, an algorithm for constraint-based causal structure learning based on domain knowledge. In particular, they suggest a feature selection to relevant variables for a given causal objective to reduce dimensionality. They include temporal order information to reduce the search space. Further, they utilize engineering knowledge to discretize continuous variables from sensors and fix causal relationships that have to exist. In contrast, our study focuses on preprocessing of raw machine log data to obtain observational data first, e.g., by applying transformation rules. Next, we include additional rules derived from domain knowledge during edge orientation. Further, we aim to apply general techniques for discretization when faced with a mixture of data. Lastly, we apply causal inference based on the learned causal graphical model.

Kühnert and Beyerer (2014) discuss techniques to detect causal structures for root cause analysis in process technology using the example of a simulated chemical stirred-tank reactor. The work focuses on handling time series of the process data, which differs from our approach and underlying assumptions. Yet, with changed assumptions, a time series-based approach might yield interesting results for the machine log data of our study as well.

Marazopoulou et al. (2016) focus on root cause analysis in an assembly line for injectors. Their study is based on sensor readings obtained from the assembly line. In a data preprocessing step, variables that contain unique keys or which have zero variance in their data are removed. The resulting set of variables is assumed to be continuous, and the PC algorithm is applied to learn the causal relationships. Within this step, domain knowledge, in the form of temporal order information, is applied during edge orientation. Further, they tune the significance level alpha using the effect strength, based on the assumption that weak dependencies are of no interest to the domain experts and cluster highly correlated variables into medoids to reduce the feature space. In contrast, in our study, we have a stronger focus on data preprocessing, incorporate discretization techniques to handle mixed data during causal structure learning, and we utilize the learned causal relationships in the causal inference step to understand key relationships.

Ye (2017) proposes a reverse engineering algorithm to identify the underlying causal structures in manufacturing systems. The proposed approach focuses on variables with binary data only. In an evaluation, the approach outperforms several Bayesian network learning techniques. In contrast, our work is more general and not restricted to learning causal structures from binary data.

Huegle et al. (2020) demonstrate how causal structure knowledge can extend existing monitoring tools in the automotive body shop assembly lines. They use the learned causal relationships between failure occurrences and quality measures of car bodies to support technical staff in time-critical failure situations highlighting potential root causes and predicting future failures. The authors mention data

Table 1 Comparison of existing work on causal reasoning in the manufacturing domain focuses on steps included in the suggested processes, from use case and input data via preprocessing and domain knowledge integration to the application of causal structures

Paper and use case	Input data	Preprocessing	Domain knowledge	Application
Li and Shi (2007) rolling process control	Product quality and process data	–	Causal objective, temporal order, engineering knowledge	Identify causal structures
Kühnert and Beyerer (2014) chemical stirred-tank reactor	Time series of process data	–	–	Root cause analysis
Marazopoulou et al. (2016) assembly line for injectors	Sensor readings	Remove variables: unique keys, zero variance	Temporal order	Root cause analysis
Ye (2017) manufacturing systems	Binary manufacturing data	–	–	Identify causal structures
Huegle et al. (2020) automotive body shop assembly line	Failure occurrences and Quality measures of car bodies	Not specified	Not specified	Failure prediction

preprocessing and domain knowledge inclusion but do not specify any concrete approaches. The learned causal model from the manufacturing machine in our work could be applied similarly. Furthermore, in our work, we elucidate data preprocessing and the application of domain knowledge during the causal structure learning process.

Beyond causal structure learning, several research works investigate the use of log data for predictive maintenance (Gutschi et al., 2019; Sipos et al., 2014; Wang et al., 2017). In this context, the preprocessing of log data follows similar steps to aggregate data within time windows and select relevant features for model creation. In general, predictive maintenance aims to avoid unexpected equipment failure to reduce downtime. In contrast to these predictions, the learned causal structures support understanding the root cause and allow for a causal effect estimation in an experimental setting applying the *do-operator*.

Theoretical background on causal reasoning

In this section, we provide a brief overview of concepts for causal reasoning. We introduce the Causal Graphical Model (CGM) (Pearl, 2009b) as a representation of causal relationships within a system of variables. We explain causal structure learning, a method to estimate the CGM from observational data, and causal inference, a formalization that allows calculating causal effects based on the estimated CGM and observational data.

Causal graphical models

In the context of causal reasoning, we follow the formal concepts of Pearl (2009b) and consider a CGM as a graph $\mathcal{G}$ defined as

$$\mathcal{G} = (\mathbf{V}, \mathbf{E}) \quad \text{with } \mathbf{V} = (V_1, \ldots, V_N) \quad \text{and } \mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}, \tag{1}$$

where $\mathbf{V}$ is a finite random set of N vertices and $\mathbf{E}$ an edge set. Each vertex V_i, $i = 1, \ldots, N$ is equivalent to an observed variables. Throughout the paper we use both terms synonymously. Further, an edge is defined as

$$(V_i, V_j) \in \mathbf{E} \qquad \text{with } i, j = 1, \ldots, N \text{ and } j \neq i. \tag{2}$$

Hence, a directed edge is denoted by

$$V_i \rightarrow V_j \qquad \text{if } (V_i, V_j) \in \mathbf{E} \text{ and } (V_j, V_i) \notin \mathbf{E}, \tag{3}$$

and an undirected edge is denoted by

$$V_i - V_j \qquad \text{if } (V_i, V_j) \in \mathbf{E} \text{ and } (V_j, V_i) \in \mathbf{E}. \tag{4}$$

This CGM $\mathcal{G}$ is assumed to be *acyclic*, i.e., we do not allow for feedback loops. Moreover, we do not allow for unmeasured confounding variables. Together with these assumptions, the CGM provides a coherent mathematical basis for causal reasoning (Pearl, 1995). The graphical d-separation criterion enables deriving conditional independence relationships through the global Markov property (Pearl, 2009b). If two vertices $V_i, V_j \in \mathbf{V}$ are not connected through an edge in $\mathcal{G}$, then they are *d-separated* in $\mathcal{G}$ by a subset $\mathbf{S}$ of the remaining vertices, called separation set. Hence, variables $V_i \in \mathbf{V}$ and $V_j \in \mathbf{V}$ are conditionally independent given a set $\mathbf{S} \subset \mathbf{V} \setminus \{V_i, V_j\}$, denoted by $V_i \perp\!\!\!\perp V_j | \mathbf{S}$. A joint distribution over the variable set $\{V_1, \ldots, V_N\}$ that satisfies the above condition is called faithful.

Causal structure learning

In most cases, the CGM $\mathcal{G}$ is unknown such that the challenge is to learn the CGM $\mathcal{G}$ from observational data. Under the assumption of the faithfulness of the distribution, the Conditional Independence (CI) information from observational data is used to reverse-engineer the CGM $\mathcal{G}$ that generated the observed data. It is well known, that several CGMs can describe exactly the same CI information and form a *Markov equivalence class* (Andersson et al., 1997; Chickering, 2002). Note, two *Markov equivalent* CGMs have the same skeleton $\mathcal{C}$, defined as

$$\mathcal{C} = (\mathbf{V}, \tilde{\mathbf{E}}) \quad \text{where } \forall\, (V_i, V_j) \in \mathbf{E} \quad (V_i, V_j), (V_j, V_i) \in \tilde{\mathbf{E}}, \tag{5}$$

and the same v-structures. V-structures are triples denoted by

$$\begin{aligned} &(V_i, V_j, V_k) \\ &\text{with } V_i, V_j, V_k \in \mathbf{V};\ i, j, k = 1, \ldots, N;\ i \neq j \neq k \\ &\text{and } V_i \rightarrow V_j, V_k \rightarrow V_j, (V_i, V_k) \notin E. \end{aligned} \tag{6}$$

The focus lies on the estimation of the equivalence class of the CGM $\mathcal{G}$ based on the corresponding probability distribution. Under Markov and faithfulness assumptions, two vertices V_i and V_j are connected through an edge in $\mathcal{G}$ if and only if they are conditionally dependent given all subsets $\mathbf{V} \setminus \{V_i, V_j\}$. This is the basis of constraint-based causal structure learning methods, e.g., the PC Algorithm, that uses CI tests to first estimate the skeleton $\mathcal{C}$ (5) and then determine the orientation of as many edges as possible, e.g., see Colombo and Maathuis (2014), Spirtes (2010), Spirtes et al. (2000), or Kalisch and Bühlmann (2007).

Causal inference

The presented properties of the CGM together with structural assignments (Peters et al., 2017; Spirtes et al., 2000) are the key to the formalization of causal inference. Traditionally, examining how a variable V_j is causally influenced by its parent V_i in $\mathcal{G}$ is built upon randomized experiments or interventions into the system under observation. Hence, observed changes in V_j cannot be associated with changes of other variables but are solely implied through $V_i \rightarrow V_j$ by the corresponding changes in V_i. An intervention on V_i changes V_i to a fixed value v_i', which graphically matches the deletion of all incoming edges of V_i in $\mathcal{G}$ transforming the observational to an experimental setup (Pearl, 2009a). This concept is formalized through Pearl's *do-operator* denoted by

$$do(V_i = v_i') \quad \text{with} \quad V_i \in \mathbf{V} \tag{7}$$

as a notion to distinguish the *observational* conditional probability denoted by

$$P(V_j = v_j | V_i = v_i') \quad \text{with} \quad V_i, V_j \in \mathbf{V}, \tag{8}$$

from the conditional *interventional* probability denoted by

$$P(V_j = v_j | do(V_i = v_i')) \quad \text{with} \quad V_i, V_j \in \mathbf{V}. \tag{9}$$

The conditional probability $P(V_j = v_j | V_i = v_i')$ describes the probability of V_j to be v_j given that $V_i = v_i'$ is observed. In contrast, the conditional interventional probability depicts the probability distribution of V_j when V_i has been set to v_i' (Pearl, 2009b). Hence, under further identifiability constraints, the CGM together with the notion for causal inference, enables to estimate causal effects through the examination of post-interventional probability distributions of V_j given $V_i = v_i'$ from purely observational data (Pearl, 1993).

Description of a real-world production stop scenario

This section describes a real-world scenario of production stops within the production process of machines that produce different products with a precision in the micrometer range twenty-four hours a day, seven days a week (Sect. 4.1). Further, we explain the data that is logged while monitoring the operation of the machines (Sect. 4.2). We relate the three introduced challenges in log data-driven causal structure learning to the production process and the machine data. Further, we mention assumptions and relaxations relevant to our model concerning the production scenario (Sect. 4.3). We want to note that specific components are not described in detail due to confidential reasons. However, these details are not necessary for the overall understanding of the use case. Instead, the more general description explicitly leaves the door open for other similar applications.

Production process

The machines subject in this study manufacture different products operating on a two-step production process. Upon receiving a new manufacturing task, the machine starts manufacturing the products, first in a ramp-up phase. The machine is calibrated during the ramp-up phase, in which manufactured products are discarded due to low quality. Once the quality of manufactured products exceeds a threshold, the machine switches into the manufacturing phase. The specified number of products as defined in the current manufacturing task is manufactured during the manufacturing phase. After finishing the current manufacturing task, the machine continues with the next manufacturing task. The entire process is monitored, e.g., to determine the quality of the products to switch from ramp-up to manufacturing

phase, and log messages are obtained and stored. The production process is interrupted when unforeseen production stops occur.

Manufacturing tasks

T denotes the set of all manufacturing tasks. A new manufacturing task $t \in T$ is issued for each product change with a set of specified configuration parameters. A task is split into two separate phases. First, the task enters its ramp-up phase t^c to calibrate the configuration parameters and assure the quality of the product. During this phase, both the machine operator and the machine itself optimize the machine settings to adjust for an accurate manufacturing result. Second, upon successful execution of the ramp-up phase, the task enters its manufacturing phase t^e, which executes the defined task and manufactures the predefined number of corresponding products from the provided input material. Accordingly, the set of tasks T can be split into subsets for the ramp-up and manufacturing phases.

Log messages

During operation, internal machine parameters are constantly monitored within different machine subsystems during the ramp-up and manufacturing phases. During monitoring, messages are logged and stored in a single event log for all machine parameters. We denote the set of all messages with M. Logging occurs continuously and event-based (Challenge II). Measurements provided through sensors are continuously logged at a sampling frequency of multiple values per second. Furthermore, event-based logging occurs either in the case that a predefined threshold is violated or in the case that a machine operator interacts with the machine. The majority of the messages are interactions of the machine operator with the machine, or non-critical issues, such as *low oil level*. There also exist messages which indicate changes in a task's phase. In particular, we distinguish between messages for the beginning of the ramp-up phase, the beginning of the manufacturing phase, and the end of the manufacturing phase. Hence, we define the following disjoint subsets of the messages M. The set of messages $M^{bc} \subset M$ contains all messages indicating the starts of the ramp-up phases, the set of messages $M^{be} \subset M$ contains all messages indicating the starts of the manufacturing phases, which also represents the ends of the ramp-up phases. Further, the set of messages $M^{ee} \subset M$ contains all messages indicating the ends of the manufacturing phases. All remaining messages are contained in $M^k \subset M$.

Production stops

A threshold violation with a respective message directly results in an unforeseen production stop for several parameters. While this informs a machine operator on a production stop and its direct trigger, it does not inform the machine operator of any causes that lead to the threshold violation.

```
class LogMessage:
    def __init__(self, time, sub_id, msg_id,
                 location=None, param=None, msg_desc=None):
        self.time                = time
        self.message_id          = msg_id
        self.location            = loc
        self.parameter_value     = param
        self.message_description = msg_desc
```

Fig. 1 Representation of a log message object with its defined mandatory and optional fields. Optional fields are marked with =None in the __init__ function

Description of machine data

The machine data logged for monitoring is semi-structured. Individual log data entries, called log message, vary in their structure. We define the following joint model for all log messages as shown in the class description for a $LogMessage$ in Fig. 1, which specifies mandatory and optional fields. The two mandatory fields, $time$, and $message\_id$ occur in all log messages and are inevitable for the transformation to observational data. The $time$ field contains a timestamp of the message's occurrence, and the $message\_id$ contains a unique identifier of the message type. Together both fields uniquely identify each log message instance. Optionally, a log message contains the three fields $location$, $parameter\_value$ and $message\_description$. The $location$ contains machine-specific location information, for example, describing the position within the machine according to components. The $parameter\_value$ contains a value from a continuous or discrete domain (Challenge III), which is sent for certain message types such as sensor readings. The $message\_description$ provides additional detail for a message. Note that both $parameter\_value$ and $message\_description$ have varying formats, depending on the machine's component and implementation.

In our example, we focus on a single machine, which logged over 40 million entries within one year. Based on the set of logged messages M, we reconstruct 25 729 tasks, cf. T, and determine over 6000 unforeseen production stops. Further, the logged messages contain 2330 distinct message types, i.e., unique message_ids. Within the context of a CGM, see (1), we will map unique message_ids to variables $\mathbf{V}$. The large number of according variables constitutes to Challenge I. Message types can be classified into four distinct classes based on domain knowledge. This classification supports the transformation to observational data, particularly for many message types, as handling them manually becomes too time-consuming. The four classes are task *configuration parameters* C, *non-critical operational messages* O, *pro-*

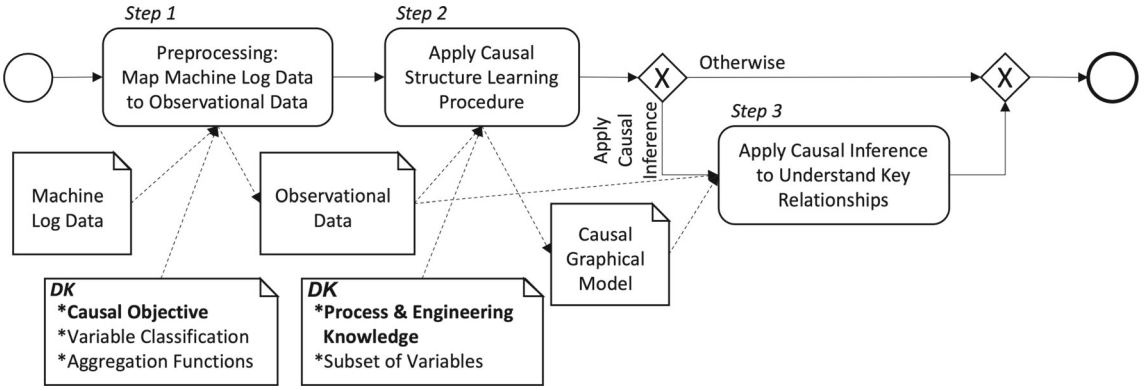

Fig. 2 Outline of the process to learn causal structures from machine log data. Rectangles with rounded corners depict process steps, while rectangles with a folded corner depict data structures. Data structures marked with *DK* represent a list of information provided by a domain expert. Bold list elements are required; other list elements are optional. Providing optional information avoids defaults and improves the quality and interpretability of results.

duction stop messages S and *continuous measurements* Q. The *configuration parameters* C subsume all message types that refer to product specifications defined for a manufacturing task. The class of *non-critical operational messages* O contains all message types that signal standard production process flow, such as the message types for changes in the task's phases or message types indicating rotation of the product. The class of *production stop messages* S contains all message types that signal an unplanned production stop during the production process, e.g., due to issues with amounts of material picked up by subsystems of the machine. Lastly, the *continuous measurements* Q subsume all message types that relate to continuously obtained measurements, such as sensor readings. We utilize this classification schema throughout the paper, e.g., when naming distinct messages, respectively variables. Note, the messages for changes in the task's phase are classified as *non-critical operational messages*.

Assumptions and relaxations

Following the theoretical background on causal reasoning (cf. Sect. 3) with the assumption of the CGM $\mathcal{G}$ to be *acyclic*, we limit the production stop scenario to exclude feedback cycles, i.e., messages cannot mutually influence each other. Further, we assume that the set of messages contains all relevant information for the production process such that we may conclude that there exist no uncollected confounding variables in the CGM. Thus, we also cannot account for outside influences such as temperature. Concerning the production process itself, we assume that the underlying mechanisms logging the data do not change within the available time period, e.g., due to software updates or hardware changes. We require this assumption to avoid violation of the independent and identically distributed (i.i.d.) requirement for observations in causal reasoning. Hence, we do not cover drift in the underlying model, respectively the CGM $\mathcal{G}$. Similarly, when generalizing the results to other machines of the precision mechanical engineering company, we assume that the same operations and processes occur. Otherwise, the causal structures have to be learned for each machine individually.

From machine log data to causal insights

In the following section, we detail our fairly general process to learn the causal structures from machine log data, as shown in Fig. 2, and apply methods of causal inference based upon the learned causal graphical model. Note, while we will explain and apply this process in the context of our use case described in Sect. 4, we would like to highlight that the process' components and concepts to be used are of general type, i.e., they are not limited to our use case and can also be adapted to analyze the log data of other applications. In particular, the information provided by the domain expert marked with DK in Fig. 2 requires adaption to correspond to the considered application. While some domain knowledge is required, see bold elements in DK data structures in Fig. 2, other domain-specific information is optional. If optional information is not provided, the process resorts to defaults, affecting the quality and interpretability of results.

In Step 1, the logged machine data is preprocessed, mapping the log messages to individual observations, to obtain observational data, addressing Challenge II (cf. Sect. 5.1). This step requires the domain experts' input of a causal objective, which guides the mapping approach. Optionally, the domain expert can classify the obtained variables from the log data and assign an aggregation function to each class of variables to improve the mapping step. The observational data is input for Step 2, the causal structure learning procedure (cf. Sect. 5.2). In this step, algorithms are applied to learn the underlying causal structures. Given the real-world data characteristics, we incorporate a variable selection and

data discretization into the causal structure learning procedure, addressing Challenge I and III. Furthermore, providing process and engineering-specific knowledge in the form of relations between classes of variables allows extending existing orientation rules to reflect domain knowledge. In the optional Step 3 causal inference to understand key relationships (cf. Sect. 5.3), the learned causal structures are input to methods of causal inference, which finally allow revealing key relationships in the considered production use case. Naturally, the findings should be validated, e.g., based on the feedback of domain experts, and general limitations have to be taken into account (cf. Sect. 5.4).

Step 1: Preprocessing: map machine log data to observational data

In this subsection, we first provide detail on our transformation procedure, which maps machine log data to observations, tackling Challenge II (Sect. 5.1.1). Within our transformation procedure, we define time windows based on the context of a domain-specific causal objective. We map the machine log messages into the time windows to extract the set of relevant variables for the CGM $\mathcal{G}$ and to transform the log messages within a time window into one observation through the application of three defined transformation rules. Second, we apply this transformation procedure to obtain the observations within our production stop scenario (Sect. 5.1.2). As a result, we obtain 25 729 observations with 1903 variables.

Transformations to derive observational data from machine log data

The transformation to derive observational data from machine log data requires the input of a causal objective based on domain knowledge. Further, optional inputs are a variable classification and specific aggregation functions. The transformation procedure uses the context of the causal objective to define time windows for individual observations. Next, all machine log messages are mapped into a time window. These mapped machine log messages are used to extract the set of N variables $\mathbf{V} = (V_1, \ldots, V_N)$ corresponding to the vertices of the CGM $\mathcal{G}$, see (1). Finally, through the definition and application of three transformation rules, the log messages within each time window are processed together with the set of variables to determine the observations. Note, if provided, variable classification and aggregation functions can replace defaults within the transformation rules. In the production process at hand, the objective to understand production stops is defined in the context of the set of manufacturing tasks T (cf. Sect. 4.1). Each manufacturing task comprises of a ramp-up phase t^c and manufacturing phase t^e. Accordingly, the following three time windows are defined for each manufacturing task.

Definition of time windows A time window δ_t is defined for a task $t \in T$. A second time window δ_t^c represents the task's ramp-up phase, while a third time window δ_t^e is defined for the task's manufacturing phase. The calculation of the time windows is based upon the timestamp $time$ of the corresponding messages drawn from the sets M^{bc}, M^{be}, M^{ee} for the task t, as shown in (10). The function $time()$ retrieves the timestamp from the selected message. Note, that for each task t, there exists exactly one message $m_t^{bc} \in M^{bc}$, one message $m_t^{be} \in M^{be}$, and one message $m_t^{ee} \in M^{ee}$. Hence, we let

$$\begin{aligned} \delta_t &:= time(m_t^{bc})\ until\ time(m_t^{ee}) \\ \delta_t^c &:= time(m_t^{bc})\ until\ time(m_t^{be}) \\ \delta_t^e &:= time(m_t^{be})\ until\ time(m_t^{ee}). \end{aligned} \tag{10}$$

Lastly, we denote the set of all time windows δ_t for all tasks T as Δ.

Mapping messages to time windows The time windows δ_t, δ_t^c or δ_t^e, see (10), for each task t are used to map all messages $m^k \in M^k$ to the task's ramp-up phase t^c or to the task's manufacturing phase t^e. Therefore, for each message m^k first, its timestamp $time$ from the log message object is selected. Next, while m^k is not mapped to a task t all time windows $\delta_t \in \Delta$ are iterated and it is checked if $time$ of m^k is within δ_t. Once the condition is met, it is specified whether the $time$ is within the corresponding time windows δ_t^c or δ_t^e. Accordingly, the log message m^k is mapped to task t corresponding to δ_t and specifically to either t^c or t^e. Note that messages outside these time windows, e.g., during general maintenance, are ignored. As a result, two lists of log message objects for each task t are returned. One list contains all messages m^k corresponding to t^c, while the other list contains all messages m^k one corresponding to t^e.

Extracting variables from messages The distinct $message\_ids$ of all log message objects make up the potential set of variables $\mathbf{V} = (V_1, \ldots, V_N)$ corresponding to the vertices of the CGM $\mathcal{G}$. Yet, given the previous transformation steps, we limit the set of variables. Therefore, we consider variables for which a $message\_id$ occurred within any time windows δ_t. Further, we distinguish for each $message\_id$ whether it occurred within the ramp-up or manufacturing phase. Note the same unique $message\_id$ is used in both phases. Therefore, we search the two lists of log message objects for each task t and extract the distinct $message\_ids$ annotated per phase. The resulting set of $message\_ids$ defines the set of variables $\mathbf{V} = (V_1, \ldots, V_N)$ for each observation.

Transformation rules Based on the set of variables $\mathbf{V}$ for all observations, we define the following three *Transformation*

rules to transform the mapped log messages per time window to independent and identically distributed observations. Hence, the three rules are applied for each task $t \in T$ and its corresponding time window $\delta_t \in \Delta$, respectively δ_t^c and δ_t^e.

Transformation rule 1 For each variable $V_i \in \mathbf{V}$, $i = 1, \ldots, N$, for which there exists only a single message in M^k within the time windows δ_t^c or δ_t^e, the value of V_i is set for the corresponding task t in the following way. If the log message object contains a parameter, the parameter's value is used as a value for V_i. Otherwise, V_i is set to 1.

Transformation rule 2 For each variable $V_i \in \mathbf{V}$, $i = 1, \ldots, N$, for which there exists no message in M^k within the time windows δ_t^c or δ_t^e, the value of V_i is set to be 0.

Transformation rule 3 For each variable $V_i \in \mathbf{V}$, $i = 1, \ldots, N$, with multiple occurrences of a message M^k within the time windows δ_t^c or δ_t^e, an aggregation function is applied to calculate the value for V_i.

For each variable $V_i \in \mathbf{V}$ a different aggregation function may be specified. If the message M^k contains a parameter, well-known aggregation functions, such as average, minimum, maximum, or last value, can be chosen. Otherwise, if the messages M^k contain no parameter, the choice of aggregation function is limited to counting occurrences or one-hot encoding the occurrence. The transformation rule three resorts to default functions, i.e., average, if the messages contain a parameter and counting if the messages contain no parameter. Yet, the defaults are overwritten if domain experts provide the optional list of aggregation functions for variables. Furthermore, in the case of a high number of variables, specifying an aggregation function for each variable becomes cumbersome. Therefore, we allow for aggregation functions to be specified for classes of variables if a variable classification is provided as input by the domain expert. Note that the choice of the aggregation function depends on the causal objective and influences the interpretability of the subsequent process steps. Hence, domain expertise is invaluable to the choice of a suitable aggregation function.

Transformation results in the production stop scenario

In the production stop scenario of the machine manufacturer, the transformation procedure from machine log data to observations results in 1903 variables with 25, 729 observations, corresponding to individual manufacturing tasks T. The variables stem from both the ramp-up and manufacturing phases. Out of the 2330 distinct message types 427 did not occur in any time window $\delta_t \in \Delta$. With close to 2000 variables choosing an aggregation function for each variable is infeasible for a domain expert. Therefore, domain experts provided aggregation functions together with a variable classification. The variable classification follows the four classes as defined in Sect. 4.2. In detail, the last value is used for *configuration parameters* C. The occurrences are counted for *non-critical operational messages* O. *Production stop messages* S are always binary, indicating whether a stop occurred or not. Lastly, *continuous measurements* Q use the mean parameter value from all messages that occurred within the current task's phase. An exemplary excerpt is shown in Table 2.

Step 2: Apply causal structure learning procedure

In this subsection, we outline the application of the causal structure learning procedure the second step of our process to learn causal structures from machine log data. First, we introduce the procedure tailored to the manufacturing domain and highlight steps that address Challenge I and III (Sect. 5.2.1). The procedure takes the observational data as input to output the learned causal graphical model. Furthermore, structured information on the process and engineering knowledge is required from the domain expert to facilitate the learning procedure. Optionally, the domain expert can specify a subset of variables to limit the size of the causal graphical model. Second, we report the learned causal structures within the production stop scenario in two settings (Sect. 5.2.2): (i) on the entire variable set and (ii) on a subset of variables selected by a domain expert.

The causal structure learning procedure

Within our work, we propose a causal structure learning procedure, see Fig. 3, based on the well-known PC algorithm (Spirtes et al., 2000), cf. Sect. 3. The procedure incorporates two steps that address the manufacturing domain-relevant Challenges I and III and augments the edge orientation step of the PC algorithm using domain knowledge.

The procedure starts with the *Select Variables* step addressing Challenge I. This step extracts a subset of variables from the observational data specified by a domain expert. Next, in the *Discretize Data* step, the observational data of the selected variables is discretized, addressing Challenge III. The skeleton graph $\mathcal{C}$ is estimated from the discretized observational data in the *Learn Skeleton* step. The resulting undirected edges, see (4), in the skeleton graph $\mathcal{C}$, see (5), are oriented using common rules of the PC algorithm and domain knowledge within the last step *Orient Edges using Domain Knowledge*.

Select variables In the first step of the procedure, a domain expert can select a subset of variables denoted by $\mathbf{V}^S \subseteq \mathbf{V}$ that is relevant for the given causal objective. The variable selection step has two goals. First, considering only a subset of variables can effectively reduce the search space within the *Learn Skeleton* step, resulting in faster execution times and poses a solution for Challenge I. Second, this step allows removing variables having similar meaning or variables known to have no relevant impact (Spirtes &

Table 2 Excerpt of the derived observational data after applying the transformation procedure from machine log data to observations. Each row marks one observational entry. Each of the columns 2–12 marks one variable. Observational data from a selection of eleven variables is presented. Note for better readability, the table header and task id are added

Task id	C_x	C_y	C_z	O_1	O_2	S_1	S_2	$Q^{\{e\}}_{speed}$	$Q^{\{c\}}_{speed}$	$Q^{\{e\}}_{number}$	$Q^{\{c\}}_{number}$
1	1020	710	11	0	0	0	1	9833	11,000	1	101
2	890	641	11	0	0	0	0	14,000	12,000	412	3
3	915	640	19	0	1	0	0	6825	13,000	172	101
...	...	...	...	...	...	...	...	...	...	...	...
25,729	1020	710	14	3	0	1	0	13,000	7800	1	404

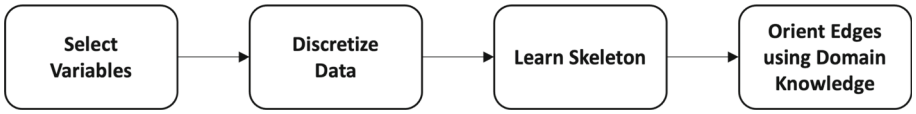

Fig. 3 Steps of the procedure to learn the causal structures, which consists of the two steps *Select Variables* and *Discretize Data* that address manufacturing domain-relevant challenges and two steps of the PC algorithm *Learn Skeleton* and *Orient Edges using Domain Knowledge*, see Section 5.2.1. Note that in this procedure the edge orientation from the PC algorithm is augmented with domain knowledge.

Scheines, 2004), which reduces the effects of noise within the dataset. The selection requires expert knowledge, as no common rule, i.e., for removing semantically dependent variables, exists (Woodward, 2016). Further, selecting variables requires caution to avoid violation of the *causal sufficiency* assumption. The subset of variables $\mathbf{V}^S$ is said to be *causally sufficient* if $\mathbf{V}^S$ incorporates all common causes or confounders that causally influence more than one variable in $\mathbf{V}^S$ (Spirtes, 2010).

Discretize data In the second step of the procedure, the observational data of variables $V \in \mathbf{V}^S$ with continuous data is discretized. Hence, as a result of this step, the observational data for all variables $V \in \mathbf{V}^S$ is assumed to be discrete. While discretization results in loss of information (Dougherty et al., 1995 , Malinsky & Danks, 2018) it allows to handle datasets with a mixture of continuous and discrete variables, addressing Challenge III. This discretization is particularly relevant for higher-dimensional datasets, where the alternative to use CI tests for mixed data (Huegle, 2021) becomes infeasible due to their long runtimes. There exist approaches that consider domain-specific discretization (Liang et al., 2004). Yet, with a larger number of variables, such an approach becomes tedious and time-consuming for any domain expert and thus infeasible in our case. Therefore, general discretization techniques, such as clustering, e.g., k-means clustering, or binning, e.g., equal-width binning, must be considered. These approaches are applied to each variable and assign the observational data of the variable to one of k representatives. The choice of k directly impacts the amount of information loss, with smaller values of k reducing the significance of the information flow (Jin et al., 2007). Yet, for large values of k, the degree of freedom within the CI test increases, demanding a higher number of observations or leading to poorer quality of the learned causal structures (Deckert & Kummerfeld, 2019). Therefore, a careful choice of the parameter k is required.

Learn skeleton In the third step of the procedure, the discrete observational data for the subset of variables $\mathbf{V}^S$ is input to the first phase of the PC algorithm, referred to as skeleton discovery. Through the repeated application of an appropriate CI tests, which is directly determined by the underlying data distribution (Dawid, 1979), the undirected skeleton graph $\mathcal{C}$, see (5), of the CGM is learned. For the discrete observational data, this step applies Pearson's χ^2 test (Pearson, 1900) as the CI test. Further, to handle high-dimensional datasets efficiently on modern hardware, this step utilizes an existing parallel implementation (Hagedorn & Huegle, 2021, Schmidt et al, 2019, Scutari, 2017).

Orient edges using domain knowledge In the final step of the procedure, the undirected edges, see (4), of the skeleton graph $\mathcal{C}$, see (5), are oriented to derive the final CGM $\mathcal{G}$, see (1). This step first applies standard orientation rules of the PC algorithm, see Colombo and Maathuis (2014), Kalisch and Bühlman (2007), or Spirtes (2010), before the application of domain knowledge according to an approach by Meek (1995). In this context, domain knowledge DK also called background knowledge, is defined as a pair $DK = (F, R)$, where $F \in \mathbf{E}$ is a set of forbidden directed edges and $R \in \mathbf{E}$ is a set of required directed edges (Meek, 1995). The original algorithm by Meek (1995) fails if the learned CGM based on the application of the standard orientation rules of the PC algorithm is not consistent with the DK, i.e., learned oriented edges violate edges in R or F. We propose to relax this consistency constraint and consider DK as recommendations, i.e., we consider R as recommended edges, and further exclude any forbidden directed edges, i.e., $F = \emptyset$. We process DK, respectively R, as follows. For each edge $E = (V_i, V_j) \in R$ check if $E \notin \mathbf{E}$ of $\mathcal{G}$. If the edge

E does not exist in $\mathcal{G}$ add $E \in R$ to a list of edge violations. Otherwise, if the edge E exists in $\mathcal{G}$ and there exists an edge $E' \in \mathbf{E}$ of $\mathcal{G}$ with $E' = (V_j, V_i)$, i.e., we have an undirected edge in $\mathcal{G}$, remove E' from $\mathcal{G}$ and close orientations (Meek, 1995) following the edge orientation rules of the PC algorithm. The list of violations is returned and allows for further investigation by the domain expert. Thus, the returned list of violations introduces a feedback loop into the causal structure learning procedure, which accounts for erroneous assumptions in the variable selection, i.e., due to violations of the semantic independence of variables (Malinsky & Danks, 2018), or erroneous assumptions in the data discretization.

Instead of relying on input from a domain expert for individual edges to define the set of recommended edges R, we define domain-specific rules to determine R. Generally, these rules can be based on temporal ordering, engineering details, or process knowledge (Liang et al., 2004). For the production process in our case study, we utilize process and engineering knowledge that is passed as input to the causal structure learning procedure to define the following rules.

Rule 1: For each variable $V_i \in \mathbf{V}$, with V_i stemming from the ramp-up phase and $V_j \in \mathbf{V} \setminus \{V_i\}$, with V_j stemming from the manufacturing phase and $i, j = 1, \ldots, N$, if there exists an edge between $V_i - V_j$ in $\mathcal{C}$, add a recommended directed edge $V_i \rightarrow V_j$ into R.

Rule 2: For each variable $V_i \in \mathbf{V}$, with V_i stemming from a configuration parameter c and $V_j \in \mathbf{V} \setminus \{V_i\}$, with V_j stemming from either a non-critical operational message o, a production stop message s or a continuous measurement m and $i, j = 1, \ldots, N$, if there exists an edge between $V_i - V_j$ in $\mathcal{C}$, add a recommended directed edge $V_i \rightarrow V_j$ into R.

Rule 1 considers process knowledge based on the context of the causal objective, recommending to orient all existing edges between variables from the ramp-up phase and variables in the production phase to orient towards the variables in the production phase. *Rule 2* considers engineering-specific information, stating that variables representing configuration parameters should not be influenced by any variables representing information obtained during the production process, regardless of the task's phase. Hence, the edge is oriented away from the configuration parameter. Note there is no particular rule covering temporal ordering.

Application of causal structure learning procedure and examination of results in the production stop scenario

The following analysis has two goals. First, we want to illustrate the application and value of learning the causal structures to understand unforeseen production downtimes in manufacturing. Second, we aim to validate our proposed process to determine accurate causal structures from machine log data. Therefore, we consider two different settings in the production stop scenario. First, in *Case (i) Application*, we take the entire set of variables $\mathbf{V}$ skipping the variable selection. Here, we demonstrate the applicability of our proposed process to derive causal structures from log data. Since validation of the process' results on the entire set of variables with over 3 million possible causal relationships is infeasible for domain experts, we consider a second case. In *Case (ii) Validation*, we let a domain expert select a validated subset $\mathbf{V}^S$ containing 11 variables. Within this subset of variables, the domain experts fully understand the mechanisms and can judge if our process correctly learned the structures, detected false positives or is missing any relevant structures. Besides, we use the resulting causal structures from $\mathbf{V}^S$ as input for methods of causal inference, to keep this illustrative example simple.

In both cases (i and ii), we discretized the sets of variables. We considered a standard equal-width binning and k-means clustering for the discretization step. We could not find a significant difference in the resulting learned causal graphs. Yet, binning showed a marginally improved result on the validated subset. Hence, for both cases, we consider the results using equal-width binning. Further, through parameter tuning, we determined a value of $k = 5$ (for equal-width binning, cf. part *Discretize Data*, Sect. 5.2.1) to produce the best results on the validated subset $\mathbf{V}^S$, which is also used as a parameter for the case of learning on the entire set of variables $\mathbf{V}$. In the subsequent skeleton learning step, we set the decision threshold for each CI test α to 0.01, which is common in practice (Colombo & Maathuis, 2014).

Case (i) Application: causal structure learning on the entire set of variables In the following, we consider the learned causal structures on the complete dataset from the globally operating machine manufacturer, omitting the step to select variables based on domain knowledge. Given the larger number of variables and a missing gold standard, a validation by a domain expert becomes infeasible, and we cannot report overall accuracy metrics for the entire set of variables. Yet, we report the accuracy of the domain knowledge-based edge orientation. Further, to understand the learned causal model, we report metrics common to describe graphical models. Besides the number of variables and learned edges corresponding to the causal relationships, we report each node's maximum and average in- and outdegree. The indegree is the number of incoming edges, whereas the outdegree describes the number of outgoing edges. Thus, the maximum and average in- and outdegree over the entire graph allow getting an understanding of the complexity of the learned causal model. For example, a low average in- and outdegree describe sparse graphical models. The parameters characterizing the learned causal relationships on the entire dataset are shown in Table 3.

The nodes in the learned graph represent the 1903 variables, which are connected by 550 edges, counting both directed, see (3), and undirected, see (4), edges. In total, 1068 nodes have no edges, meaning no causal relationship to any

Table 3 Parameters describing the learned causal model on the entire machine dataset. The parameters consist of the number of variables N, the number of learned edges $|\mathbf{E}|$, the average indegree $avg(deg^-(\mathbf{V}))$, the maximum indegree $\triangle^-(\mathbf{V})$, the average outdegree $avg(deg^+(\mathbf{V}))$ and the maximum outdegree $\triangle^+(\mathbf{V})$

Parameter	N	$\lvert\mathbf{E}\rvert$	$avg(deg^-(\mathbf{V}))$	$\triangle^-(\mathbf{V})$	$avg(deg^+(\mathbf{V}))$	$\triangle^+(\mathbf{V})$
Value	1903	550	1.21	4	1.12	4

other node. The maximum indegree $\triangle^-(\mathbf{V})$ and maximum outdegree $\triangle^+(\mathbf{V})$ are both four and the average indegree $avg(deg^-(\mathbf{V}))$ is slightly higher than the average outdegree $avg(deg^+(\mathbf{V}))$ with a factor of 1.21 compared to 1.12.

Concerning the domain knowledge-based edge orientation, we found that nine edges of the 550 edges have a wrong orientation according to the set R. The edges in R stem entirely from *Rule 1*. Hence, there is no violation of *Rule 2*. Further, four of these edges have been oriented following the approach of Meek (1995). Regarding the unforeseen production stops, 11 unique production stop messages, respectively variables, exist in the data. For seven of these variables, a total of eight causal relationships have been identified.

While there is little error according to the domain knowledge-based edge orientation, full validation of all existing and non-existing edges is out of scope for any domain expert. Additionally, we assume that many nodes without any causal relationship indicate that several variables obtained from the log data have little relevance to the production process. Yet, these variables might introduce additional noise to the model, impacting its overall quality. Hence, fully relying on the learned causal structures within the entire set of variables is not advisable. Thus, the identified causal relationships for the production stop messages should only be considered an indication for further investigation of root causes, for example, through a careful selection of relevant variables by a domain expert, including the identified variables.

Case (ii) Validation: causal structure learning on a domain-specific variable selection For the second case, a domain expert selected a subset $\mathbf{V}^S$ of 11 variables for which the underlying causal mechanisms are known. In this context, the goal is to derive a relevant and comprehensible sub-problem that can be evaluated through domain expertise. The variables' classification, data type, and description are provided in Table 4.

The 11 variables represent three task configuration parameters, two non-critical operational messages, two different production stop messages, and four continuous measurements. Note, the two variables for the measurements, shown in Table 4 have a realization during the ramp-up phase Q^c_X and the manufacturing phase Q^e_X, with X being either *number* or *speed*. Overall, the variables cover three data types: categorical, binary, or discretized. Note that discretized means a continuous variable has been discretized using binning. The categorical variables of the product

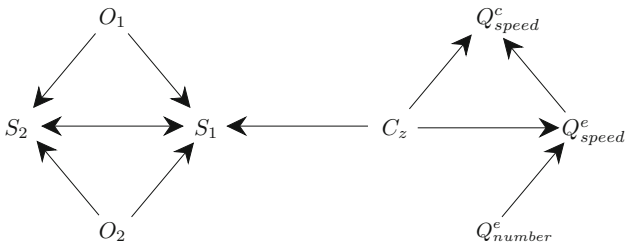

Fig. 4 The learned causal graphical model, resulting from the application of process steps 1 and 2. Note, that a subset of 11 variables was selected from the entire machine dataset and that the phase of the measurements, ramp-up (^{c}) or execution (^{e}) is added at the end of the variable

dimensions C_x, C_y, C_z have between two to five different categories. The binary variables are either zero or one. A zero indicates that the operational message or production stop did not occur in the corresponding observation. In contrast, one indicates that the operational message or production stop happened in the corresponding observation.

The resulting causal graphical model is shown in Fig. 4. Note, C_x and C_y are causally dependent on each other and have, similar to Q^c_{number}, no relationship to any other variable. Therefore, they are omitted in Fig. 4. Further, the learned causal relationships suggest an influence of the operational messages O_1, O_2, as well as C_z on the unforeseen production stops S_1, S_2. Both product rotations within the machine and the material's and product's thickness can cause internal machine subsystems to pick up the wrong amount of material for the finishing step. Thus, causing a production stop due to no material or too much material in the finishing step. In this case, the relationship between the two production stops is plausible, too. The causal relationship of the product thickness C_z between the machine speeds $Q^{\{e/c\}}_{speed}$, as well as the relationship of the number of produced products Q^e_{number} to the manufacturing speed Q^e_{speed}, is also confirmed. The domain expert questions only the causal relationship of the machine speed during manufacturing Q^e_{speed} to the machine speed during the ramp-up phase Q^c_{speed}. This particular edge is marked by the algorithm as a violation of *Rule 1* and is considered to be a wrongly detected edge. We believe the wrong direction to be caused by the discretization of the measurements. Overall, the solution obtained was confirmed by the experts.

Table 4 The 11 variables contained in the validated dataset, including their classification, their type and a description of its meaning. Note that the bottom two variables Q_{speed} and Q_{number} have two representations one in each task's phase, ramp-up (^{c}) and execution (^{e})

Variable	Classification	Type	Description
C_x	Configuration parameter	Categorical	Product length
C_y	Configuration parameter	Categorical	Product width
C_z	Configuration parameter	Categorical	Product thickness
O_1	Non-critical operational message	Binary	Product rotated clockwise
O_2	Non-critical operational message	Binary	Product rotated anti-clockwise
S_1	Production stop	Binary	No material in finishing step
S_2	Production stop	Binary	too much material in finishing step
$Q_{speed}^{\{e/c\}}$	Measurement	Discretized	Machine speed
$Q_{number}^{\{e/c\}}$	Measurement	Discretized	Number of produced products

The validated learned CGM is a starting point to support a machine operator to identify causes for the production stops S_1, S_2, e.g., rotations of the product during production O_1, O_2 or the product thickness C_z. In contrast to commonly applied classification methods where the most relevant variables are used for root cause analysis, e.g., see Chien and Chuang (2014) or Oliveira et al. (2022), the knowledge about causal structures not only provides the opportunity to detect sequences of root causes but also distinguishes between associative and causal variables (Pearl, 1995). For example, the product thickness C_z as a common confounder induces an associative dependence of the machine speed Q_{speed}^c and production stops S_1. Hence, the relevance of Q_{speed}^c is increased within the classification approach, although there exists no causal effect of Q_{speed}^e on S_1.

Due to the restrictive selection of variables, sequences of influences beyond the subgraph cannot be identified, e.g., causes for O_1 or O_2, such that domain experts should consider a larger set of relevant variables in practice to investigate also influences for O_1, O_2.

Step 3: Apply causal inference to understand key relationships

The framework of CGMs together with the *do-operator* (cf. (7)) allows for an estimation of causal effects in an experimental regime on the basis of observational data (Peters et al., 2017). The application of the *do-operator* provides answers to questions, such as, *What is the impact of a machine change?*, which might not be affordable in real-world settings due to production process interruption or unavailability of an experimental setup.

Furthermore, an examination of the causal relationships avoids wrong assumptions and decisions based on conditional probabilities. For example, consider the following scenario of a domain expert investigating the causes of the production stop S_1, i.e., $S_1 = 1$. In this setting, a domain expert aims to decrease the probability $P(S_1 = 1)$, by changing values of possible causes. Let us assume the machine operator considers that the machine speed during manufacturing Q_{speed}^e impacts the occurrence of production stop S_1 as indicated by a classification approach (see Case (ii) of Sect. 5.2.2). In this setting, the machine operator consults the data, inspecting conditional probabilities of $S_1 = 1$ given any of the five categories of Q_{speed}^e during manufacturing, following (8), as displayed in Table 5. The data indicates that using a machine speed within the range represented by category $k = 1$ yields the lowest probability of an occurrence of the production stop $S_1 = 1$.

In this example, the *do-operator* enables to calculate the post-interventional conditional probabilities of S_1, i.e., $P(S_1 = 1|do(Q_{speed}^e = k))$, following (9), to examine the causal effect of an intervention on Q_{speed}^e to 1. Based on the result, the machine operator could judge if changing the machine speed results in the desired reduction in the occurrence of the production stop $S_1 = 1$. To calculate the respective probabilities, we utilize the R-package *causaleffect* (Tikka & Karvanen, 2017), which applies rules to determine a formula to calculate the respective probabilities under a given intervention. Note, the *causal.effect* function operates on a Directed Acyclic Graph (DAG). Hence, for the learned graph depicted in Fig. 4, we select one of the two represented DAGs. For the case of $S_1 = 1$ the conditional probability given an intervention on S_{speed}^e is given by $P(S_1 = 1|do(Q_{speed}^e = 1))$. Applying the formula determined by the R-package *causaleffect* when intervening on $Q_{speed}^e = 1$ coincides with the unconditioned probability of $P(S_1 = 1) = 13.4\%$. Thus, for our example, changes to the machine speed Q_{speed}^e do not influence S_1 and based on this assumption, the machine operator would have applied impractical changes to the machine setting. Affirmatively, the learned causal graphical model does not contain any causal relationship between the two variables Q_{speed}^e and S_1.

Table 5 The conditional probabilities for the occurrence of the unique stopper $S_1 = 1$, when either selecting a distinct machine speed during execution $Q^e_{speed} = k$ or not selecting any, represented by unconditioned. The machine speed was discretized into five categories $k = 0, 1, \ldots, 4$ using equal-width binning

	Unconditioned	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$P(S_1 = 1 \mid Q^e_{speed} = k)$	13.4%	10.6%	8.6%	15.3%	13.3%	13.3%

Discussion, validation, and limitations

In the following discussion, we point out necessary considerations concerning generalizability and limitations that need to be mentioned.

Discussion and validation

We analyzed a real-world use case from a globally operating precision mechanical engineering company to show the applicability of our customized process for causal structure learning and causal inference based on machine log data. The results verified that the proposed process to learn causal structures from log data could support machine operators to, e.g., identify the root causes of unexpected production downtimes. In particular, the machine operator consults the causal structures to determine the variables, which have edges pointing to the production stop message monitored at production downtime. Thus, the search space to remove the production stop is reduced, and time is saved. Note, as the time of experienced machine operators is valuable and limited such kind of automated data-driven support can be highly beneficial.

Application in practice

For an efficient integration into the workflow of a machine operator, we suggest an integration of the learned causal structures into an existing monitoring solution, see Huegle et al. (2020). Based on the two learned causal models (cf. Sect. 5.2.2), we suggest a careful selection of variables when using causal structure learning in practice. On the one hand, to avoid noise, i.e., when using all variables, on the other hand, to avoid missing influences, i.e., when selecting to restrictive. Further, we suggest visually presenting only the relevant selection of the causal model for the occurring production stop to provide a focus for the machine operator. Extending this monitoring solution with the capability for causal inference further strengthens the support for the machine operator. The integration of causal inference is beneficial for inexperienced machine operators, as it avoids drawing false conclusions (cf. Sect. 5.3).

Transferability to other domains

We see the potential to apply the proposed process in similar production settings, e.g., automotive production. In these production settings, monitoring systems are in place, and similar messages are logged. Further, using domain knowledge to specify a causal objective in combination with a set of definitions, similar to the ones defined in Sect. 5.1, e.g., based on time or location constraints, e.g., by utilizing the $location$ information of a $LogMessage$, allows applying the transformation rules to derive sound observational data. To sketch a concrete example from automotive production, consider the causal objective to understand the reasons for defective cars in the context of the car assembly. The car is worked on in different assembly stations within the car assembly at specific points in time (Huegle et al., 2020). Thus, analogously to our proposed process, time windows can be detected for the car's presence in each station, and all monitoring log messages are mapped accordingly. Hence, the observations can be constructed using the same set of transformation rules, and causal structures can be learned following our proposed process.

Yet, it remains to investigate if the generalized approaches to apply the same aggregation function for a category of messages or a single discretization approach yield acceptable results in these settings. In the context of discretization, we see the potential for an automatic selection of the most appropriate discretization technique for each continuous variable. Such a step requires a validated subset to determine the impact on the learned causal structures and a generalization to the remaining variables, e.g., based on the variable categories. Further, it remains to investigate if the transformation rules remain applicable in domains in which constraints other than the time dimension are used to derive sound observational data. The applicability of the transformation rules may become a limiting factor for the transferability to other domains.

Methodological limitations

While the proposed process yields appropriate results in our use case, it is worth examining methodological limitations that need to be considered when applying causal structure learning and causal inference. First, note that the assumptions of methods for causal structure learning are quite restrictive. In this context, it is important to check whether the particular dataset satisfies the required preconditions such as causal sufficiency, i.e., no latent confounding variables. Note that there exist a variety of extensions to the PC algorithm that allow the application of causal structure learning under weakened

assumptions, e.g., the FCI algorithm in case of violated causal sufficiency (Spirtes, 2010). In this context, it is worth mentioning that our approach may serve as the basis to capture time-dependent causal effects through the implementation of the methodological extensions for time-related causal graphical models with respective algorithms.

Second, besides the theoretical restrictions on the dataset, the accuracy of our approach is strongly influenced by the trustworthiness of the incorporated domain knowledge. In particular, incorrect domain knowledge within the process of causal structure learning from machine log data, see Fig. 2, may yield not only unreliable observational data but also wrong causal structures and hence an incorrect causal inference. Moreover, while discretization improves the interpretability of both causal structures and causal effects, a wrong technique may preserve relevant causal relationships.

Third, while adequate within our use case, data quality is a well-known problem in practice and requires an additional step for data cleaning, e.g., the imputation of missing values. In the context of machine log data, changes of both the logging technique and systematic changes within the machine, such as modifications during the observation period, may yield changes within the underlying causal mechanism and hence inconsistent learned causal structures.

Therefore, both the implementation of the proposed process for causal structure learning and the interpretation of the results should be made carefully. For more information on challenges of causal structure learning in practice, see the practical guide of Malinsky and Danks (2018).

Conclusion

We proposed a process to learn causal structures in the context of a discrete manufacturing production process, which comes with domain-specific challenges. The causal relationships are learned based on data logged for monitoring the manufacturing machines during operation. Using real-world data from a globally operating machine manufacturer, we showed how to apply all single process steps to derive the causal graphical model, which we utilize to estimate causal effects in an experimental regime. Our example showed how to provide data-driven decision support to assist domain experts in avoiding wrongly assumed influences of unforeseen production stops.

Our proposed process integrates domain knowledge at different steps to increase the quality and interpretability of the results. First, transformation rules are defined to extract sound observations from the log data (cf. Challenge II). Second, domain experts are encouraged to select relevant variables to reduce the search space and noise in high-dimensional settings (cf. Challenge I). To handle mixed data, we suggest discretizing the data (cf. Challenge III) before applying parallel constraint-based causal structure learning algorithms. Further, we extend the rules commonly applied during the algorithm's edge orientation. Note this procedure and the basic methods applied in our approach could also address the causal reasoning from machine log data in similar applications. In this context, we point out limitations and constraints for generalizability that need to be considered.

In future work, the methodological limitations can be addressed, by considering causal structure learning methods with weaker assumptions, e.g., the FCI algorithm (Spirtes, 2010), through detailed studies on the impact of discretization (Deckert & Kummerfeld, 2019), and investigating model drift concerning systematic changes within machines. Additionally, future work needs to develop a sophisticated validation method to verify entire learned causal models with domain experts. This validation method allows receiving a quantifiable judgment of the correctness of the applied techniques in the given context of the machine log data.

Funding Open Access funding enabled and organized by Projekt DEAL.

Code availability Code with a synthetic data sample available on GitHub: https://github.com/ChristopherSchmidt89/csl-downtimes

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

Andersson, S. A., Madigan, D., & Perlman, M. D. (1997). A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics, 25*(2), 505–541.

Boudjelida, A. (2019). On the robustness of joint production and maintenance scheduling in presence of uncertainties. *Journal of Intelligent Manufacturing, 30*(4), 1515–1530.

Chen, K. S., & Huang, M. L. (2006). Performance measurement for a manufacturing system based on quality, cost and time. *International Journal of Production Research, 44*(11), 2221–2243.

Chickering, D. M. (2002). Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research, 2*(3), 445–498.

Chien, C. F., & Chuang, S. C. (2014). A framework for root cause detection of sub-batch processing system for semiconductor manufacturing big data analytics. *IEEE Transactions on Semiconductor Manufacturing, 27*(4), 475–488.

Colombo, D., & Maathuis, M. H. (2014). Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research, 15*(116), 3921–3962.

Davis, J., Edgar, T., Graybill, R., et al. (2015). Smart manufacturing. *Annual Review of Chemical and Biomolecular Engineering, 6*(1), 141–160.

Dawid, A. P. (1979). Conditional independence in statistical theory. *Journal of the Royal Statistical Society: Series B (Methodological), 41*(1), 1–31.

Deckert, A. C., & Kummerfeld, E. (2019). Investigating the effect of binning on causal discovery. In *2019 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2019*. IEEE Computer Society, pp. 2574–2581.

Dougherty, J., Kohavi, R., & Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., pp. 194–202.

Du, S., Lv, J., & Xi, L. (2012). A robust approach for root causes identification in machining processes using hybrid learning algorithm and engineering knowledge. *Journal of Intelligent Manufacturing, 23*(5), 1833–1847.

Gutschi, C., Furian, N., Suschnigg, J., et al. (2019). Log-based predictive maintenance in discrete parts manufacturing. *Procedia CIRP, 79*, 528–533.

Hagedorn, C., & Huegle, J. (2021). GPU-accelerated constraint-based causal structure learning for discrete data. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*. Society for Industrial and Applied Mathematics, pp. 37–45.

Hernán, M. A., & Robins, J. M. (2020). *Causal inference: What If*. Chapman & Hall/CRC.

Huegle, J. (2021). An information-theoretic approach on causal structure learning for heterogeneous data characteristics of real-world scenarios. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, IJCAI'21, pp. 4891–4892.

Huegle, J., Hagedorn, C., & Uflacker, M. (2020). How causal structural knowledge adds decision-support in monitoring of automotive body shop assembly lines. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, IJCAI'20, pp. 5246–5248, demos.

Jin, R., Breitbart, Y., & Muoh, C. (2007). Data discretization unification. In *Proceedings - IEEE International Conference on Data Mining*. IEEE Computer Society, pp. 183–192.

Kalisch, M., & Bühlmann, P. (2007). Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research, 8*, 613–636.

Khatab, A. (2018). Maintenance optimization in failure-prone systems under imperfect preventive maintenance. *Journal of Intelligent Manufacturing, 29*(3), 707–717.

Kühnert, C., & Beyerer, J. (2014). Data-driven methods for the detection of causal structures in process technology. *Machines, 2*(4), 255–274.

Le, T. D., Hoang, T., Li, J., et al. (2019). A fast PC algorithm for high dimensional causal discovery with multi-core PCs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 16*(5), 1483–1495.

Li, J., & Shi, J. (2007). Knowledge discovery from observational data for process control using causal Bayesian networks. *Institute of Industrial Engineers Transactions, 39*(6), 681–690.

Li, Z., Wang, Y., & Wang, K. (2020). A data-driven method based on deep belief networks for backlash error prediction in machining centers. *Journal of Intelligent Manufacturing, 31*(7), 1693–1705.

Liang, S. Y., Hecker, R. L., & Landers, R. G. (2004). Machining process monitoring and control: The state-of-the-art. *Journal of Manufacturing Science and Engineering, 126*(2), 297–310.

Liu, J., Chang, Q., Xiao, G., et al. (2012). The costs of downtime incidents in serial multistage manufacturing systems. *Journal of Manufacturing Science and Engineering, 134*(2), 1–10.

Liu, Q., Dong, M., Lv, W., et al. (2019). Manufacturing system maintenance based on dynamic programming model with prognostics information. *Journal of Intelligent Manufacturing, 30*(3), 1155–1173.

Maathuis, M., Drton, M., Lauritzen, S., et al. (2018). *Handbook of graphical models* (1st ed.). CRC Press Inc.

Malinsky, D., & Danks, D. (2018). Causal discovery algorithms: A practical guide. *Philosophy Compass, 13*(1), 1–11.

Marazopoulou, K., Ghosh, R., & Lade, P. et al. (2016). Causal discovery for manufacturing domains. Retrieved from https://arxiv.org/abs/arXiv:1605.04056

Meek, C. (1995) Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., UAI, pp. 403–410.

Mobley, R. K. (2002). *An introduction to predictive maintenance. Plant engineering* (2nd ed.). Butterworth-Heinemann.

Nikula, R. P., Karioja, K., Leiviskä, K., et al. (2019). Prediction of mechanical stress in roller leveler based on vibration measurements and steel strip properties. *Journal of Intelligent Manufacturing, 30*(4), 1563–1579.

e Oliveira, E., Miguéis V. L., & Borges, J. (2021). Understanding overlap in automatic root cause analysis in manufacturing using causal inference. *IEEE Access, 10*, 191–201.

e Oliveira, E., Miguéis, V. L., Borges, J. (2022). Automatic root cause analysis in manufacturing: An overview & conceptualization. *Journal of Intelligent Manufacturing, 2022,* 1–18.

Pearl, J. (1993). Comment: Graphical models, causality and intervention. *Statistical Science, 8*(3), 266–269.

Pearl, J. (1995). Causal diagrams for empirical research. *Biometrika, 82*(4), 669–688.

Pearl, J. (2009). Causal inference in statistics: An overview. *Statistics Surveys, 3*, 96–146.

Pearl, J. (2009). *Causality: models, reasoning, and inference* (2nd ed.). Cambridge University Press.

Pearson, K. F. (1900). X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 50,*(302), 157–175.

Peters, J., Janzing, D., & Schölkopf, B. (2017). *Elements of causal inference: foundations and learning algorithms*. Adaptive computation and machine learning seriesMIT Press.

Qin, W., Zha, D., & Zhang, J. (2020). An effective approach for causal variables analysis in diesel engine production by using mutual information and network deconvolution. *Journal of Intelligent Manufacturing, 31*(7), 1661–1671.

Rodríguez, A. R., Bernal de Lázaro, J. M., Prieto-Moreno, A., et al. (2019). An approach to robust fault diagnosis in mechanical systems using computational intelligence. *Journal of Intelligent Manufacturing, 30*(4), 1601–1615.

Rokach, L., & Hutter, D. (2012). Automatic discovery of the root causes for quality drift in high dimensionality manufacturing processes. *Journal of Intelligent Manufacturing, 23*(5), 1915–1930.

Schmidt, C., Huegle, J., & Bode, P. et al. (2019). Load-balanced parallel constraint-based causal structure learning on multi-core systems for high-dimensional data. In *Proceedings of Machine Learning Research*, vol 104. PMLR, pp. 59–77.

Scutari, M. (2017). Bayesian network constraint-based structure learning algorithms: Parallel and optimized implementations in the bnlearn r package. *Journal of Statistical Software, Articles, 77*(2), 1–20.

Sipos, R., Fradkin, D., & Moerchen, F., et al. (2014). Log-based predictive maintenance. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, pp. 1867–1876.

Spirtes, P. (2010). Introduction to causal inference. *Journal of Machine Learning Research, 11*, 1643–1662.

Spirtes, P., & Scheines, R. (2004). Causal inference of ambiguous manipulations. *Philosophy of Science, 71*(5), 833–845.

Spirtes, P., & Zhang, K. (2016). Causal discovery and inference: Concepts and recent methodological advances. *Applied Informatics, 3*(1), 1–28.

Spirtes, P., Glymour, C., & Scheines, R. (2000). *Causation, prediction, and search* Adaptive computation and machine learning (2nd ed.). MIT Press.

Sun, Y., Qin, W., Zhuang, Z., et al. (2021). An adaptive fault detection and root-cause analysis scheme for complex industrial processes using moving window KPCA and information geometric causal inference. *Journal of Intelligent Manufacturing, 32*(7), 2007–2021.

Tikka, S., & Karvanen, J. (2017). Identifying causal effects with the R package causal effect. *Journal of Statistical Software, Articles, 76*(12), 1–30.

Wang, J., Li, C., Han, S., et al. (2017). Predictive maintenance based on event-log analysis: A case study. *IBM Journal of Research and Development, 61*(1), 121–132.

Woodward, J. (2016). The problem of variable choice. *Synthese, 193*(4), 1047–1072.

Wuest, T., Irgens, C., & Thoben, K. D. (2014). An approach to monitoring quality in manufacturing using supervised machine learning on product state data. *Journal of Intelligent Manufacturing, 25*(5), 1167–1180.

Wuest, T., Weimer, D., Irgens, C., et al. (2016). Machine learning in manufacturing: Advantages, challenges, and applications. *Production and Manufacturing Research, 4*(1), 23–45.

Ye, N. (2017). A reverse engineering algorithm for mining a causal system model from system data. *International Journal of Production Research, 55*(3), 828–844.