

causalAssembly: Generating Realistic Production Data for Benchmarking Causal Discovery

Konstantin Göbler

*Technical University of Munich, Germany
Robert Bosch GmbH*

KONSTANTIN.GOEBLER@TUM.DE

Tobias Windisch

University of Applied Sciences Kempten

TOBIAS.WINDISCH@HS-KEMPTEN.DE

Mathias Drton

*Technical University of Munich, Germany
Munich Center for Machine Learning (MCML)*

MATHIAS.DRTON@TUM.DE

Tim Pychynski

Steffen Sonntag

Martin Roth

Robert Bosch GmbH

TIM.PYCHYNSKI@DE.BOSCH.COM

STEFFEN.SONNTAG@DE.BOSCH.COM

MARTIN.ROTH2@DE.BOSCH.COM

Editors: Francesco Locatello and Vanessa Didelez

Abstract

Algorithms for causal discovery have recently undergone rapid advances and increasingly draw on flexible nonparametric methods to process complex data. With these advances comes a need for adequate empirical validation of the causal relationships learned by different algorithms. However, for most real and complex data sources true causal relations remain unknown. This issue is further compounded by privacy concerns surrounding the release of suitable high-quality data. To tackle these challenges, we introduce *causalAssembly*, a semisynthetic data generator designed to facilitate the benchmarking of causal discovery methods. The tool is built using a complex real-world dataset comprised of measurements collected along an assembly line in a manufacturing setting. For these measurements, we establish a partial set of ground truth causal relationships through a detailed study of the physics underlying the processes carried out in the assembly line. The partial ground truth is sufficiently informative to allow for estimation of a full causal graph by mere nonparametric regression. To overcome potential confounding and privacy concerns, we use distributional random forests to estimate and represent conditional distributions implied by the ground truth causal graph. These conditionals are combined into a joint distribution that strictly adheres to a causal model over the observed variables. Sampling from this distribution, *causalAssembly* generates data that are guaranteed to be Markovian with respect to the ground truth. Using our tool, we showcase how to benchmark several well-known causal discovery algorithms.

Keywords: Causal discovery, benchmarking, production data, distributional random forest

1. Introduction

Causal discovery is the process of learning causal relations among variables of interest from data. In recent years, the area has seen numerous theoretical and algorithmic developments (for an overview see, e.g. [Glymour et al., 2019](#); [Heinze-Deml et al., 2018](#); [Maathuis et al., 2019](#); [Vowels et al., 2022](#); [Zanga et al., 2022](#)). Causal structure learning is also gaining traction in industrial domains (compare e.g. the review in [Vuković and Thalmann, 2022](#)), where knowledge about causal relations forms the basis for downstream operations such as root-cause analysis ([Budhathoki et al., 2022](#)).

Despite the recent advances and expanded application areas, it remains challenging to provide adequate empirical validation of the causal relationships inferred by causal discovery algorithms (Gentzel et al., 2019; Eigenmann et al., 2020). Ground truth causal relationships are unknown for most real-world examples that involve complex data sources. In addition, privacy concerns often hinder the release of high-quality data. As a result, research papers proposing novel procedures often work with simplistic simulation setups that tend to have limited generalizability to real-world settings (Huegle et al., 2021; Reisach et al., 2021). When applied to real data and assessed by domain experts, results are often sobering (Heinze-Deml et al., 2018; Constantinou et al., 2021).

To help address these challenges we introduce `causalAssembly`, a semisynthetic data generation tool that leverages extensive domain knowledge and real production data to form a ground truth. The assembly line that originates the data is composed of multiple production stations where individual components are joined together through automated manufacturing processes. Each individual manufacturing process highly depends on the state of the preceding processes as well as on the raw components added in earlier stations. Due to the intricate and often nonlinear nature of the physical processes—including, e.g., the interaction of machine state, process control settings and component state, complex causal relationships arise across the entire assembly line. In the manufacturing plant, processes are computer-monitored, and the resulting measurements are stored throughout production.

We select a subset of production stations from the assembly line, each of which contains individual processes with known causal relationships. We determine these relationships by consulting with domain experts and carefully studying the physical processes that take place at each station. However, our process knowledge currently does not extend to relationships between processes. We show that the extensive domain knowledge of the individual processes, together with the line structure, gives rise to a causal ordering that agrees with that of the unknown ground truth layered graph. Consequently, we apply established feature selection strategies (e.g. Bühlmann et al., 2014) to prune the corresponding complete graph implied by this causal ordering. The resulting causal graph is treated as the appropriate ground truth in `causalAssembly`.

Next, we need to enable sampling from a ground truth distribution that factorizes according to the ground truth causal graph. This is achieved with a synthetization step that learns conditional distributions indicated by the ground truth graph via distributional random forests (DRF) (Cevic et al., 2022; Gamella et al., 2022). Combining the conditionals yields a joint distribution that strictly obeys a causal model for the observed variables. Thus, the distribution of semisynthetic data sampled with `causalAssembly` is guaranteed to be Markov with respect to the ground truth graph. We want to note that strong-faithfulness (see Uhler et al., 2013) can not be guaranteed to hold in the sampled data which may affect consistency results of some causal discovery algorithms.

Our contributions. We summarize our main contributions below:

- Using domain knowledge and real production data, we propose a semisynthetic data generation pipeline `causalAssembly`. Implemented as a Python library¹, the tool allows one to generate data of varying dimension and complexity in order to benchmark causal discovery algorithms.
- The proposed procedure to synthesize real data is general and can in principle be applied to any real dataset with full or partial domain knowledge. The data generation procedure assures resulting data to be Markovian under causal sufficiency.

1. <https://github.com/boschresearch/causalAssembly>

- The proposed semisyntheticization of real data offers a way to release high-quality data to the public that would otherwise be inaccessible due to privacy concerns. To employ this procedure, knowledge regarding a causal order will be a necessary requirement.
- We provide first benchmark results of popular causal discovery algorithms including the PC algorithm (Spirites and Glymour, 1991), DirectLiNGAM (Shimizu et al., 2011), NOTEARS (Zheng et al., 2018), GraN-DAG (Lachapelle et al., 2020), and SCORE (and its scalable extension DAS) (Rolland et al., 2022; Montagna et al., 2023).

In the remainder of the paper, we first address identifiability concerns and provide an overview of related work (Section 2). Section 3 describes the data generation pipeline, and Section 4 showcases benchmarking results. Section 5 gives a concluding discussion of our work. Supplementary Sections A-D provide additional information regarding data, algorithms and metrics used, as well as additional benchmark results.

2. Identifiability and common benchmarking strategies

2.1. General background

For natural number $p \in \mathbb{N}$, let $[p] := \{1, \dots, p\}$. Defining the vertex set as $V = [p]$, let $G = (V, E)$ be a directed acyclic graph (DAG) with edge set $E \subset V \times V$. The vertex set V indexes a collection of jointly distributed random variables $(X_v)_{v \in V}$. For $S \subseteq V$, we write $X_S := (X_s)_{s \in S}$ for the subcollection of variables indexed by S . We often write $v \rightarrow w$ or $X_v \rightarrow X_w$ instead of (v, w) to indicate the existence of an edge from v to w in G . A node $v \in V$ is a *parent* of another node $w \in V$ if $v \rightarrow w \in E$. The set of all parents of w in G is denoted by $pa_G(w)$. A *directed path* from v to w in G is a sequence of nodes v_1, \dots, v_k starting at $v_1 = v$ and ending at $v_k = w$ with subsequent nodes connected by edges $v_i \rightarrow v_{i+1} \in E$. If there exists a directed path from v to w , then v is an *ancestor* of w and w a *descendant* of v . The sets of all ancestors and all descendants of node w in G are denoted $an_G(w)$ and $de_G(w)$, respectively and $w \notin an_G(w) \cup de_G(w)$. When it is clear which graph G is being considered, we simply write $pa(w)$, $de(w)$ or $an(w)$.

The DAG G permits causal relationships among the random variables, and we say that X_v is a direct cause of X_w if $v \rightarrow w$. Furthermore, X_v may be an indirect cause of X_w if there is a directed path from v to w ; we refer to Pearl (2009); Peters et al. (2017) for more background. Each DAG G on V gives rise to a statistical model for the joint distribution P_X of the random vector $X = (X_v)_{v \in V}$. Here, the edges in G encode the permitted conditional dependence relations among the variables (see e.g. Drton and Maathuis, 2017; Lauritzen, 1996). Any distribution that satisfies the conditional independence constraints implied by G is said to be *Markov* with respect to G . In other words, the joint density of X factors into a product of conditional densities as $p(x) = \prod_{v \in V} p(x_v | x_{pa_G(v)})$ with $p(x_v | x_\emptyset) = p(x_v)$. The conditional independence relations holding in all such factorizing distributions can be obtained using the graphical criterion of *d-separation* (Pearl, 2009). A distribution P_X is *Markov* and *faithful* with respect to the underlying DAG G if the conditional independence relations in P_X correspond exactly to d-separation relations in G . Different DAGs may share the same set of conditional independence statements. Such DAGs are said to be Markov equivalent. All equivalent DAGs form a Markov equivalence class (MEC) (see e.g. Spirites et al., 1993).

2.2. Layered DAGs

Let $G = (V, E)$ be a DAG, and let $\mathcal{L} := (V_1, \dots, V_K)$ be a partition of V into K ordered subsets. Then $L = (G, \mathcal{L})$ is a *layered DAG* if for every edge $(v, w) \in E$ with nodes $v \in V_s, w \in V_t$ with $s \neq t$, we have $s < t$ (see, e.g., [Healy and Nikolov, 2002](#)). In this case, the elements of \mathcal{L} are the *layers* of L , and each layer V_s induces a subgraph $G[V_s]$ of G , which we denote by L_s and refer to as the layer-induced subgraph.

A permutation $\pi : [p] \rightarrow [p]$ is a causal ordering for the layered DAG $L = (G, \mathcal{L})$ if (i) $\pi(v) < \pi(w)$ for all (v, w) with $w \in \text{de}_G(v)$, and (ii) $\pi(v) < \pi(w)$ for all (v, w) such that $v \in V_s$ and $w \in V_t$ for layers with $s < t$. Such an ordering respects both the ordering given by directed paths in G and the ordering of the layers in \mathcal{L} . There always exist a causal ordering but it need not be unique. The set of all causal orderings of a layered DAG L is denoted by Π_L .

Proposition 1 *Let $L = (G, \mathcal{L})$ and $L' = (G', \mathcal{L})$ be two layered DAGs that share the same vertex set V and the same partition $\mathcal{L} = (V_1, \dots, V_K)$. If $L_s = L'_s$ for all $s \in [K]$, then $\Pi_L = \Pi_{L'}$.*

The proof can be found in Section B. Let $L = (G, \mathcal{L})$ be a layered DAG with vertex set V and edge set E . For any causal ordering $\pi \in \Pi_L$, we define a completed layered DAG $L^\pi = (G^\pi, \mathcal{L})$ by adding to L all edges $v \rightarrow w$ with $v \in V_s$ and $w \in V_t$ for $s < t$. So, the edge set of L^π is

$$E^\pi := E \cup \bigcup_{s < t} V_s \times V_t.$$

Clearly, L^π is a super-DAG of L in the sense of E being a subset of E^π . Often we have to consider all nodes up to a certain layer. Thus, we define $V_{1:s} := \cup_{i=1}^{s-1} V_i$ for $s \in [K]$. Note that $V_s \cap V_{1:s} = \emptyset$.

2.3. Identifiability of the causal structure

In general, the DAG G is not identifiable from P_X alone, and learning the graph or its MEC from data requires assumptions. Under the Markov and faithfulness conditions on P_X , G can be recovered up to its MEC. Prominent examples of algorithms that are based on this assumption are the PC algorithm ([Spirtes et al., 1993](#)) and Greedy Equivalence Search (GES) ([Chickering, 2003](#)). Under additional assumptions, we may not only recover the MEC but G itself. Note that the graphical model associated with G can also be expressed in terms of a structural causal model (SCM) ([Peters et al., 2017](#)). That is, in the absence of latent confounding, if P_X is Markov with respect to G then there exist independent noise variables $\epsilon = (\epsilon_1, \dots, \epsilon_p)$ and measurable functions g_v such that $X_v = g_v(X_{\text{pa}(v)}, \epsilon_v)$ for all $v \in [p]$. In this framework, if all the g_v are linear and all ϵ_v follow non-Gaussian distributions then the causal graph is identifiable from data ([Shimizu, 2022](#)). In a different vein, identifiability of G is also achieved under an additive noise model (ANM) ([Hoyer et al., 2008](#)). In other words, as long as all g_v are non-linear as defined in [Hoyer et al. \(2008\)](#), the noise ϵ_v may follow any distribution given that it is additive. A generalization to the post-nonlinear model is proposed by [Zhang and Hyvärinen \(2009\)](#). In contrast, linear SCMs with Gaussian noise terms are not identifiable. However, identifiability can be achieved in this setting under further assumptions such as equal noise variances ([Peters and Bühlmann, 2014](#)).

2.4. Additive noise models for synthetic data generation

Comprehensive assessment of methods for learning causal relations is challenging due to the scarcity of real data where true causal relations are known. Hence, the focus often lies on simulated synthetic

data. In order to accommodate non-linearities and also to ensure identifiability of the true DAG, ANMs are often employed for this purpose. However, synthetic data generation can leave unintended traces hinting at the true causal relations among the variables. In particular, [Reisach et al. \(2021\)](#) demonstrate that for commonly used simulation setups involving ANMs, irrespective of the choice of the functions g_v , ordering nodes by increasing marginal variance tends to agree with the causal ordering of the ANM. The authors introduce *varsortability*, a score that measures the degree of concordance between causal orderings of the ground truth DAG and the order of increasing marginal variance. When varsortability equals one then all directed paths lead from lower variance nodes to higher variance nodes. This is connected to the aforementioned identifiability results for linear SCMs with equal variance ([Chen et al., 2019](#)). Many recently proposed continuous optimization-based structure learning algorithms have been evaluated in simulations based on ANMs (e.g., [Zheng et al., 2018](#); [Ng et al., 2020](#); [Lachapelle et al., 2020](#); [Yu et al., 2021](#); [Zhang et al., 2022, 2023](#)). The resulting high varsortability on the raw data scale contributes to often staggering performances as the methods may inadvertently make use of variance order. In real systems, on the other hand, measurement scales are essentially irrelevant for the causal ordering. Thus, standardization would render data scale non-informative. Yet, remnants of high varsortability with distinctive covariance patterns may remain even after standardization ([Reisach et al., 2021, 2023](#)). Ultimately, all synthetic data generation processes are vulnerable to a lack of portraying common themes observable in practice.

Evaluating the performance of algorithms is a key factor that impacts which research directions the field is gravitating towards. This can be seen, for instance, in the case of continuous optimization-based structure learning. Following the development of NOTEARS ([Zheng et al., 2018](#)), numerous extensions have been proposed in short succession (see [Vowels et al., 2022](#), for an overview). However, when evaluated outside of ANMs, performance often drops notably ([Constantinou et al., 2021](#)). Therefore, there is a clear necessity for more robust evaluation techniques in the field of causal discovery as outlined in [Eigenmann et al. \(2020\)](#); [Gentzel et al. \(2019\)](#). This motivates the present work which seeks to support benchmarking of causal discovery by providing tools for semisynthetic data generation that draw on partial ground truth and real data from a manufacturing plant.

2.5. Related work

Focusing on observational data, the available tools for empirical illustration and benchmarking of causal discovery algorithm can be broadly classified into three tiers: (1) Real datasets with expert-knowledge driven ground truth, such as the cause-effect pairs of [Mooij et al. \(2016\)](#) and data pertaining to already well understood systems in biology ([Marbach et al., 2012](#); [Replogle et al., 2022](#); [Sachs et al., 2005](#)). (2) Synthetic datasets generated based on well-established parameterizations ([den Bulcke et al., 2006](#); [Schaffter et al., 2011](#); [Scutari, 2010](#)). (3) Data generated using functional specifications; see, e.g., [Chen and Fayek \(2023\)](#); [Huegle et al. \(2021\)](#). Our proposed semisynthetic data generator takes a middle ground between (1) and (2), leveraging expert knowledge to establish a causal order while relying on real data to circumvent the need for explicit parameterizations.

3. Semisynthetic data generator

3.1. Process knowledge

Our real-world data come from an assembly line at a highly automated manufacturing plant. We focus on a selected subset of production stations at which press-in and staking processes are executed.

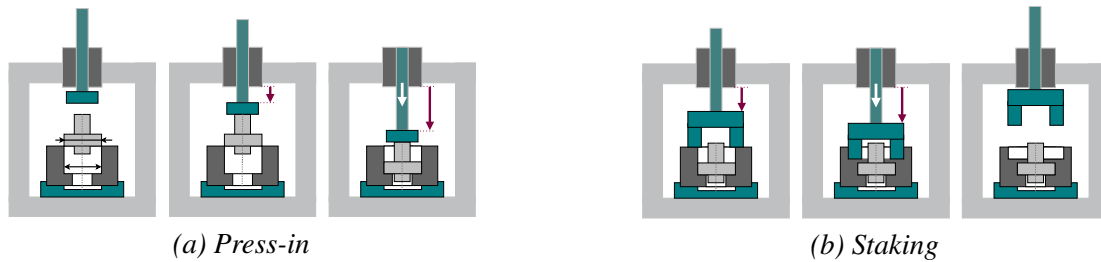


Figure 1: Illustration of the phases of a press-in and staking process. In press-fitting (a1) the tool moves downward axially until it contacts the valve (a2) and pushes it into a slightly smaller bore until its shoulder reaches the axial block position (a3).

In the following, we give a brief overview of these processes. A more in depth discussion of the relations among the central features of the physical procedures is given in supplementary Section A.1.

Press-in: In each fitting process, two components are joined in a pressing machine to form a mechanically strong and sometimes even leakage tight connection between the components. An inner cylindrical component (e.g., a valve, axle or pin) is pushed into a slightly smaller bore of a larger component. This is done by applying a mechanical force high enough to overcome the required friction force. After the friction force is exceeded, the inner component will move into the bore. The pressing machine stops as soon as a certain predefined press-in force is exceeded for the first time.

Staking: A staking process is an extension of a press-in process and has the objective to fixate the inner component in its final position by permanently deforming outer component materials. This is achieved by applying a large enough force to the tool to create a mechanical stress-strain state in the material resulting in permanent plastic material deformation.

Figure 1 depicts these two process steps in a schematic manner. During both steps the required forces and tool displacements are measured continuously by process control units. We extract all relevant numeric measurements for each process, and together with the mapping detailed in Section A.1 they build the foundation for our data generation scheme.

3.2. Modeling domain knowledge

Let $L^0 = (V, E^0, \mathcal{L})$ be the true and unknown layered DAG that encapsulates within as well as between process relations. The associated random variables $(X_v)_{v \in V}$ represent the features extracted from the $K = 10$ manufacturing processes that are carried out sequentially. These processes are lined up across five production stations. A user of `causalAssembly` will see which variables belong to each of these five stations. Each station accommodates two of the production processes (Fig. 2). In total, $p = |V| = 98$ features for $n = 15.581$ parts are collected. The features distribute across production stations as follows: *Station 1* with $|V_1 \cup V_2| = 6$, *Station 2* with $|V_3 \cup V_4| = 34$, *Station 3* with $|V_5 \cup V_6| = 16$, *Station 4* with $|V_7 \cup V_8| = 26$, and *Station 5* with $|V_9 \cup V_{10}| = 16$.

Each individual process we focus on is well understood, and we are able to state causal relations among the variables in V_s . These causal relations are considered to be complete and correct for the individual processes. We map the process knowledge into process DAGs $G_s^* = (V_s, E_s)$ for $s \in [K]$. The graph union $G_1^* \oplus \dots \oplus G_K^*$ together with the ordered partitioning implied by the node sets of the process DAGs gives rise to the layered DAG $L^* = (G_1^* \oplus \dots \oplus G_K^*, \mathcal{L})$ with $\mathcal{L} = (V_1, \dots, V_{10})$.

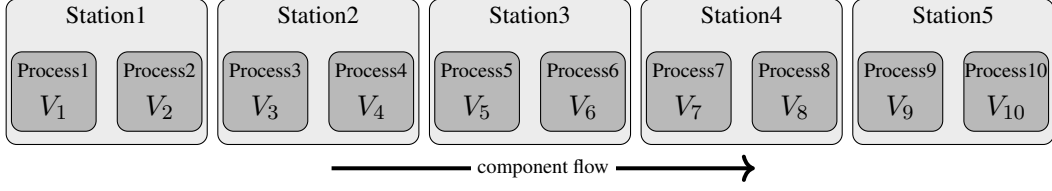


Figure 2: Illustration of the production line with five process stations each containing two successive processes. The first station prepares components while station two and four carry out staking tasks. The remaining stations perform press-in tasks.

Note that L^* and L^0 are both layered DAGs on the same vertex set V with the same layering \mathcal{L} . More importantly, the layer-induced subgraphs coincide, i.e., $L_s^* = L_s^0$ for all $s \in [10]$.

Unfortunately, due to their complex physical nature, available process knowledge does not extend to how the K processes influence each other. In other words, the edge sets of L^* and L^0 likely differ. However, Proposition 1 suggests that the process knowledge we gathered puts us in a favorable situation: Irrespective of whether there exist edges between layers, the set of all possible causal orderings is determined solely by the layers themselves, i.e. the process graphs. The complete layered DAG L^π for any $\pi \in \Pi_{L^*}$ is a super-DAG of the true layered DAG L^0 . Consequently, we may use common variable selection techniques from regression to prune L^π and infer the layered ground truth DAG L^0 (Bühlmann et al., 2014, Section 2.5).

3.3. Sparse additive models (SpAM) for cross-process pruning

We perform the pruning step by regressing each variable X_v onto its parent variables X_k , $k \in pa_{L^\pi}(v)$. The resulting active sets of predictors determine the final edge set, forming the estimate \hat{L} . Unless stated otherwise, the parent sets considered below are obtained from L^π . When constructing L^π we add edges between the nodes from distinct processes V_s and V_t , and leave the layer-induced subgraphs as they are. Similarly, we only consider edges between processes for pruning. To prune, we use the flexible nonparametric framework of SpAMs (Ravikumar et al., 2009). In a SpAM, the regression function for a variable X_v with v in layer V_t , is modeled as

$$f_v(x_{pa(v)}) = \mathbb{E}(X_v | X_{pa(v)} = x_{pa(v)}) \approx \sum_{k \in pa(v)} f_{v,k}(x_k).$$

Sparsity is induced through a functional group-lasso penalty based on $\|f_{v,k}\|_2 := \sqrt{\mathbb{E}(f_{v,k}(X_k)^2)}$, the $L^2(P_{X_k})$ norm of component $f_{v,k}(X_k)$. For some $\lambda \geq 0$, the resulting population-level minimization problem can be stated as:

$$\min_{f_{v,k} \in \mathcal{F}_{v,k}, k \in pa(v)} \left\{ \frac{1}{2} \mathbb{E} \left(X_v - \sum_{k \in pa(v)} f_{v,k}(X_k) \right)^2 + \lambda \sum_{k \in V_{1:t}} \|f_{v,k}\|_2 \right\}.$$

Depending on the function classes $\mathcal{F}_{v,k}$, the minimization in SpAM becomes a convex program. Indeed, in practice, we express each function as $f_{v,k}(x_k) = \sum_{l=1}^{p_{vk}} \psi_{vkl}(x_k) \beta_{vkl}$, where $\{\psi_{vkl}\}_{l=1}^{p_{vk}}$ is a set of cubic splines. Let $\Psi_{vk}(x_k)$ be the $1 \times p_{vk}$ row vector of evaluations of the ψ_{vkl} , and gather the β_{vkl} in the coefficient vector $\theta_{vk} = (\beta_{vk1}, \dots, \beta_{vkp_{vk}})^T$. Then $f_{v,k}(x_k) = \Psi_{vk} \theta_{vk}$. For estimation from data, we solve the empirical version of the SpAM problem, with cross-validation for

tuning parameter selection. The set $\{(k, v) : k \in V_{1:t}, \hat{\theta}_{vk} \neq 0\} \cup pa_{L_s^*}(k)$ then marks the indices of the parent variables of X_v remaining after pruning. Algorithm 1 outlines our procedure.

Algorithm 1 SpAM cross-process learning

Input: Data $\mathcal{D} \subseteq \mathbb{R}^p$, completed layered DAG L^π , regularization parameter $\lambda \geq 0$

Output: Edge set $E_{\text{across}} \subset V \times V$.

$E_{\text{across}} \leftarrow \{\};$

for $t = 2, \dots, K$ **do**

 Form a causal ordering π_t of the layer-induced subgraph L_t^π ;

for v along π_t **do**

 Obtain $(\hat{\theta}_{vk})_{V_{1:t}}$ through the following minimization problem:

$$\min_{\theta_{vk}, k \in pa(v)} \left\{ \frac{1}{2|\mathcal{D}|} \sum_{x \in \mathcal{D}} (x_v - \sum_{k \in pa(v)} \Psi_{vk}(x_k) \theta_{vk})^2 + \lambda \sum_{k \in V_{1:t}} \sqrt{\frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} (\Psi_{vk}(x_k) \theta_{vk})^2} \right\}$$

$E_{\text{across}} \leftarrow E_{\text{across}} \cup \{(k, v) : k \in V_{1:t}, \hat{\theta}_{vk} \neq 0\}$

end

end

For some variables X_v with v in a layer V_t , $t \in \{2, \dots, K\}$, the process experts are able to state functional relations, up to noise. Denote $P_v^* := pa_{L^*}(v) \cap V_t$. Functional process knowledge takes the form $X_v \approx f_v^*(X_{P_v^*})$. To incorporate such knowledge, we alter Algorithm 1 by inserting the known function. The optimization problem is then

$$\min_{\theta_{vk}, k \in pa(v)} \left\{ \frac{1}{2|\mathcal{D}|} \sum_{x \in \mathcal{D}} (x_v - f_v^*(x_{P_v^*}) - \sum_{k \in V_{1:t}} \Psi_{vk}(x_k) \theta_{vk})^2 + \lambda \sum_{k \in V_{1:t}} \sqrt{\frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} (\Psi_{vk}(x_k) \theta_{vk})^2} \right\}$$

Note that Algorithm 1 admits a natural way to also incorporate cross-process domain knowledge simply by adding the corresponding edge to E or dropping it from the corresponding parent set.

Figure 3 illustrates the process graph \hat{L} after the pruning step using Algorithm 1. In the absence of latent confounding the sparsistency property of the SpAM procedure suggests that asymptotically we recover the true layered DAG L^0 . Even in the presence of confounding the synthetization procedure illustrated in the next section assures that data will be ‘‘Markovian’’. From here on we consider the layered DAG \hat{L} as ground truth in `causalAssembly`.

3.4. Synthetization

To ensure that the semisynthetic data come from a distribution that factorizes according to the ground truth DAG \hat{L} , we explicitly estimate the conditional distributions under the factorization implied by the independence constraints in \hat{L} . To that end, we employ distributional random forests (DRFs) (Cevid et al., 2022) to obtain estimates $\hat{P}_X^{\text{DRF}}(X_v | X_{pa(v)})$, $v \in V$, along a causal ordering of \hat{L} .

DRFs build on random forests (Breiman, 2001) where the original splitting criterion is replaced with a distributional metric (in our case the maximal mean discrepancy (MMD) statistic (Gretton et al., 2006)) in order to ensure homogeneity in each of the tree’s leaf nodes. These give rise to neighborhoods of relevant training data points described by corresponding weighting functions.

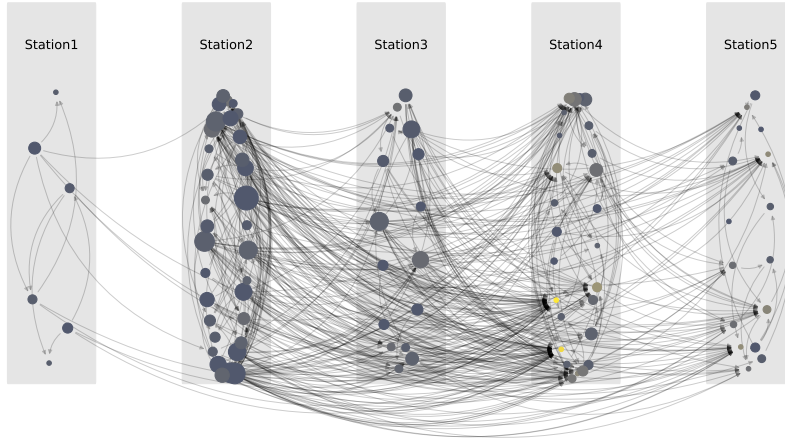


Figure 3: Assembly line ground truth after edges have been learned between processes using SpAM. We depict production stations, not the processes they are decomposed in. Node size increases with the number of out-edges. Node color gets brighter with the number of in-edges. In terms of sparsity, the assembly line ground truth graph accounts for around 10.2% of all possible connections.

The average of the tree-specific weighting functions can then be used to estimate the conditional distributions. The following paragraphs give a brief overview of the DRF procedure.

DRF growing Let L be a layered DAG. Assume we have grown N trees $\mathcal{T}_1, \dots, \mathcal{T}_N$ and denote $\mathcal{L}_j(x)$ the set of training data points that are assigned to the same leaf node as the test data point x in tree \mathcal{T}_j . [Cevid et al. \(2022\)](#) define the weighting function by the average of the tree-specific weighting functions i.e. $w_x(x_i) = 1/N \sum_{j=1}^N \mathbb{1}(x_i \in \mathcal{L}_j(x)) / |\mathcal{L}_j(x)|$. Let n be the sample size of the training data. Estimating the conditional distribution $P(X_v | X_{pa_L(v)} = x)$ boils down to assigning weights to the point mass $\delta_{x_{i_v}}$ at x_{i_v} in the empirical distribution of X_v , i.e.

$$\hat{P}(X_v | X_{pa_L(v)} = x) = \sum_{i=1}^n w_x(x_i) \cdot \delta_{x_{i_v}}.$$

Given $\hat{P}_X^{\text{DRF}}(X_v | X_{pa(v)})$ we sample a new dataset $\mathcal{D}_{\text{synth}} \subset \mathbb{R}^p$ from the production line as outlined in [Algorithm 2](#). By construction, $\mathcal{D}_{\text{synth}}$ is drawn from the estimated joint distribution \hat{P}_X^{DRF} and is guaranteed to be Markov with respect to the ground truth layered DAG \hat{L} .

[Gamella et al. \(2022\)](#) first proposed this approach using [Sachs et al. \(2005\)](#) data. By construction, it resolves confounding issues in the real data at the expense of \hat{L} potentially containing more cross-process edges than L^0 .

Remark 2 *We emphasize the crucial role of the acquired process knowledge. Although the synthesization procedure is technically capable of fitting DRFs to any configuration of conditional distributions given a sufficient sample size, domain knowledge plays a pivotal role in ensuring the generated data closely resembles the real data. To illustrate this point, we conduct experiments in [Section C.2](#) where we assume no prior domain knowledge and instead rely solely on structure learning. Upon applying the semisynthetic procedure to a resulting process graph, we observe that in a majority of experiments, the causal discovery algorithm that initially generated the process graph*

Algorithm 2 Sampling from the causal DRFs**Input:** Finite dataset $\mathcal{D}_{\text{original}} \subset \mathbb{R}^p$, (layered) DAG L , number of samples n to be generated**Output:** A dataset $\mathcal{D}_{\text{synth}} \subset \mathbb{R}^p$ of size n $\mathcal{D}_{\text{synth}} \leftarrow \emptyset$;Form a causal ordering π of L ;**while** $|\mathcal{D}_{\text{synth}}| < n$ **do** **for** v along π **do** **if** $pa_L(v) = \emptyset$ **then** | sample $\hat{x}_v \in \mathbb{R}$ via smooth bootstrapping from $\{x_v \in \mathbb{R} : x \in \mathcal{D}_{\text{original}}\}$; **else** | sample $\hat{x}_v \in \mathbb{R}$ from $\hat{P}_X^{\text{DRF}}(X_v | X_{pa_L(v)} = \hat{x}_{pa_L(v)})$; **end** $\mathcal{D}_{\text{synth}} \leftarrow \mathcal{D}_{\text{synth}} \cup \{\hat{x}_v\}$; **end****end**

structure tends to exhibit superior performance. As we demonstrate in Section 4, this is not the case for our semisynthetic data generator. We investigate, whether learning the cross-process edges via SpAM makes learning these edges easier. However, Figure 12 in Section C.2 suggests that this is not the case. The combination of extensive process knowledge and highly flexible nonparametric techniques employed in `causalAssembly` renders it virtually impossible for any causal discovery algorithm to unfairly "game" graph recovery results.

To accommodate the analysis of data with varying dimensionality, we provide the option to isolate and generate data for any of the 5 production stations that form the natural subunits of the assembly line. To avoid creating unintended confounding we refit the DRFs and apply Algorithm 2 to each production station independently. Lastly, we want to stress that SpAMs are merely a pruning method. Their application in `causalAssembly` is not related to ANMs that have been discussed in Section 2.4.

3.5. Semisynthetic data description

To demonstrate that the semisynthetic "Markovian" data obtained from `causalAssembly` preserves key characteristics of the real data, we calculate the two-sample Kolmogorov-Smirnov (KS) statistic (Kolmogorov, 1933) between all variables in the real and a semisynthetic dataset. Figure 4 shows kernel density estimates of variables with largest and smallest KS statistics among all non-source nodes. Even in unfavorable cases where the absolute difference between the marginal distributions is largest, the semisynthetic data only shows moderate deviations from the real data. To further illustrate the similarity of the semisynthetic data to the real data, we depict pairwise correlation agreement in Figure 11 in the Appendix.

In Figure 5, we depict bivariate relationships between selected direct causes on the x-axis and their effects on the y-axis according to the ground truth DAG \hat{L} . Variables were selected within stations in (a) and (b) and between stations in (c). The size of the parent sets for the effect variables are 1, 3, 6 for (a), (b), and (c), respectively. Naturally, with increasing parent set size bivariate

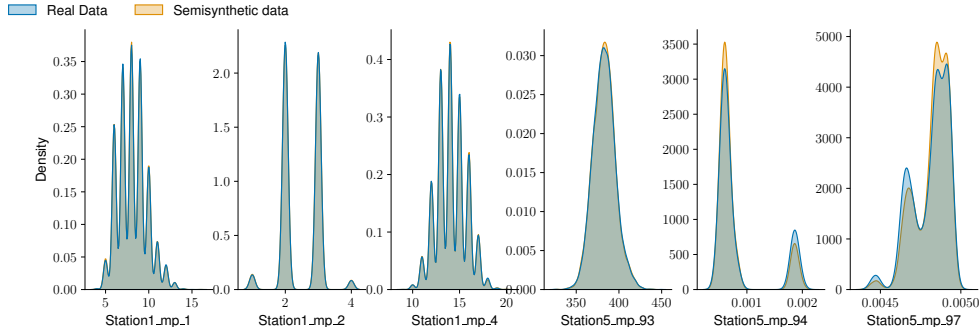


Figure 4: Kernel density plots (upper panel) of the same variables in the real (blue) and semisynthetic (yellow) data. Selection of those variables is based on comparing the highest (first three) and lowest (last three) agreement in terms of Kolmogorov-Smirnov statistic.

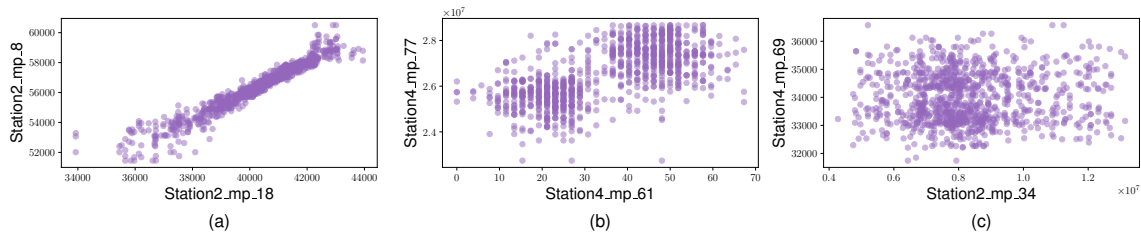


Figure 5: Bivariate scatter plots of selected nodes to showcase the different types of bivariate patterns to expect in data generated with `causalAssembly`. All node pairs are causally linked with x-axis variables being parents of y-axis variables. The number of parents of each y-axis variable varies from one parent in (a), three parents in (b) and six parents in (c).

relationships become harder to discern. While (a) depicts a clear linear relationship between the selected measurements, the relationships in (b) and (c) are masked by the remaining parent variables.

4. Benchmark results

As a proof of concept, we present first causal discovery benchmark results. In order to make sure that privacy concerns are respected, we first apply the semisynthetization procedure to the real data from the manufacturing plant. We then sample once from the joint distribution in `causalAssembly`. The resulting data is used for benchmarking as shown below. We present further benchmark results based on the individual production stations in Section D.2.

The purpose of the study we report on here is to demonstrate the versatility of the data generation scheme. A more elaborate set of experiments comprising varying sample size and adjusting hyperparameters will be taken up in future work. The PC algorithm does not output a DAG. Thus, to compare results additional harmonization steps are necessary. We implement two strategies *CPDAG-transform* and *average-dag* outlined in Section D.1. As the results following these strategies do not seem to differ substantially, we report the *average-dag* results and refer to Appendix D for the remaining experiments. A brief overview of the algorithms and metrics is provided in Section D.1.

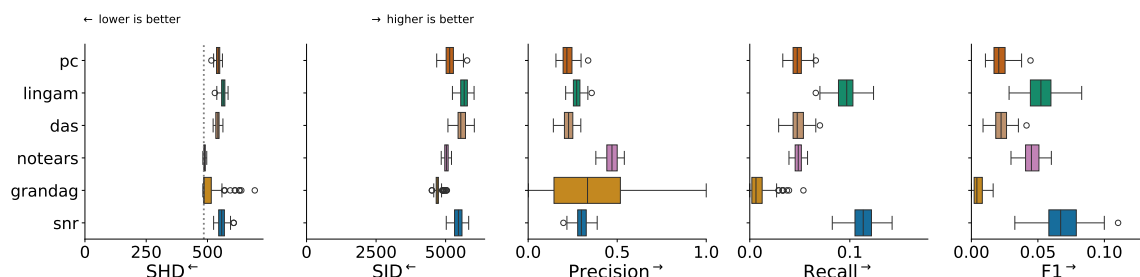


Figure 6: Chosen metrics for assessing causal discovery of the full assembly line ($|V| = 98$) using 100 simulation runs on a sample size of $n = 5000$. The dashed line corresponds to the SHD between the empty graph and ground truth indicating the number of edges in the ground truth graph. Note that we use DAS rather than SCORE on the full line due to its improved scaling to higher dimensions.

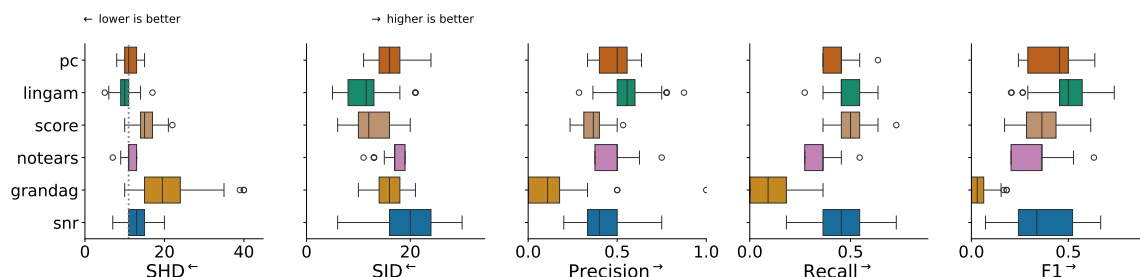


Figure 7: Chosen metrics for assessing causal discovery outcomes on Station 3 ($|V_5 \cup V_6| = 16$) using 100 simulation runs on a sample size of $n = 500$. The dashed line corresponds to the SHD between the empty graph and ground truth indicating the number of edges in the ground truth graph.

We standardize the data sampled in each of the 100 simulation runs before structure learning and report the average varsortability in Table 4 in Section D. The sortnregress (snr) routine serves as a reference point, as on standardized data it corresponds to choosing some ordering at random and regressing each node on its ancestors (*random regress*) (see also Reisach et al., 2021).

Figures 6 and 7 illustrate the benchmark results for the full line and Station 3, respectively. We focus on Station 3 because its results differ significantly from those of the full line. The benchmark results for the remaining four process stations are displayed in Section D.

Regarding the full line benchmark results in Figure 6, the default methods fail to beat an empty graph in terms of structural Hamming distance (SHD). Overall, NOTEARS performs best in terms of SHD and SID, but this is partly due to its tendency to output very sparse graphs, as evidenced by low Recall scores. Linear methods such as DirectLiNGAM and the PC algorithm with standard partial correlation tests perform similarly to *random regress*. GraN-DAG exhibits higher variability in results, with highly variable precision and very low recall and F1 scores. Inspecting the results from Station 3 depicted in Figure 7, we observe that most selected algorithms outperform random regress, especially in terms of structural intervention distance (SID) and F1 score. DirectLiNGAM exhibits the best performance across most metrics, while GraN-DAG again struggles with precision, recall, and F1. The PC algorithm and SCORE perform similarly, and NOTEARS fails to beat *random regress* in this case.

We emphasize that these results are based on out-of-the-box implementations of the algorithms, with no attempt to tune them to the data. Additionally, all the considered methods ignore the prior knowledge implied by the layered structure of the processes. We expect that causal discovery tools that can efficiently incorporate the layered structure will achieve better performance. Furthermore, tuning the hyperparameters of the algorithms and using statistical methods that are better equipped to handle complex nonlinear relationships is likely to yield better results.

While limited, this first benchmark study reveals that standard off-the-shelf methods struggle with data generated by `causalAssembly`. This largely corroborates findings from empirical studies, which show that the learning accuracy reported in the literature on synthetic data overestimates performance on more complex datasets (Constantinou et al., 2021; Rios et al., 2022).

5. Conclusion and Limitations

Empirical validation is key in understanding how causal discovery algorithms perform outside their laboratory settings. In this work, we leverage a complex dataset comprising measurements from a highly automated assembly line in a manufacturing plant, for which we are able to put together partial ground truth causal relationships on the basis of a detailed study of the underlying physics. Based on this dataset, we use the associated ground truth and develop a versatile synthetization pipeline based on DRFs that helps overcome common real world data problems such as lack of ground truth, confounding, and privacy concerns. At the same time, by preserving the real data characteristics as much as possible while making sure that the observational distribution is Markov with respect to the obtained ground truth, we overcome the issue of purely synthetic data often being too simplistic and not generalizable to real world settings. The resulting data generator can be used to benchmark causal discovery algorithms based on complex production data. We demonstrate this by providing initial benchmarking results for some well-known causal discovery algorithms.

An important limitation is that currently `causalAssembly` supports the generation of continuous data only. However, the methodology presented is more general and supports mixed data. We plan to include discrete features by selecting a wider range of processes in future releases of `causalAssembly`. Another limitation comes from the fact that ground truth relations are only available within and not across processes. In future releases we plan to address this by continuously validating cross-process edges together with process experts. Lastly, it would be interesting to take advantages of opportunities to involve interventional samples.

Acknowledgments

We thank the anonymous reviewers and Jens Ackermann for helpful comments and insights. The research for this project was partly conducted by authors employed at Robert Bosch GmbH.

References

- Steen A. Andersson, David Madigan, and Michael D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *Ann. Statist.*, 25(2):505–541, 1997. ISSN 0090-5364,2168-8966. doi: 10.1214/aos/1031833662. URL <https://doi.org/10.1214/aos/1031833662>.
- Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. doi: 10.1023/a:1010933404324. URL <https://doi.org/10.1023/a:1010933404324>.
- Kailash Budhathoki, Lenon Minorics, Patrick Bloebaum, and Dominik Janzing. Causal structure-based root cause analysis of outliers. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 2357–2369. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/budhathoki22a.html>.
- Peter Bühlmann, Jonas Peters, and Jan Ernest. CAM: causal additive models, high-dimensional order search and penalized regression. *Ann. Statist.*, 42(6):2526–2556, 2014. ISSN 0090-5364,2168-8966. doi: 10.1214/14-AOS1260. URL <https://doi.org/10.1214/14-AOS1260>.
- Domagoj Cevic, Loris Michel, Jeffrey Näf, Peter Bühlmann, and Nicolai Meinshausen. Distributional random forests: Heterogeneity adjustment and multivariate distributional regression. *Journal of Machine Learning Research*, 23(333):1–79, 2022. URL <http://jmlr.org/papers/v23/21-0585.html>.
- Jarry Chen and Haytham M. Fayek. The structurally complex with additive parent causality (scary) dataset. In *2nd Conference on Causal Learning and Reasoning*, 2023.
- Wenyu Chen, Mathias Drton, and Y. Samuel Wang. On causal discovery with an equal-variance assumption. *Biometrika*, 106(4):973–980, 2019. ISSN 0006-3444. doi: 10.1093/biomet/asz049. URL <https://doi-org.eaccess.tum.edu/10.1093/biomet/asz049>.
- David Maxwell Chickering. Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3:507–554, 2003. ISSN 1532-4435. doi: 10.1162/153244303321897717. URL <https://doi.org/10.1162/153244303321897717>.
- Diego Colombo and Marloes H. Maathuis. Order-independent constraint-based causal structure learning. *J. Mach. Learn. Res.*, 15:3741–3782, 2014. ISSN 1532-4435,1533-7928.
- Anthony C. Constantinou, Yang Liu, Kiattikun Chobtham, Zhigao Guo, and Neville K. Kitson. Large-scale empirical validation of Bayesian network structure learning algorithms with noisy data. *International Journal of Approximate Reasoning*, 131:151–188, 2021. doi: 10.1016/j.ijar.2021.01.001. URL <https://doi.org/10.1016/j.ijar.2021.01.001>.
- Tim Van den Bulcke, Koenraad Van Leemput, Bart Naudts, Piet van Remortel, Hongwu Ma, Alain Verschoren, Bart De Moor, and Kathleen Marchal. SynTRen: A generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7(1), 2006. doi: 10.1186/1471-2105-7-43. URL <https://doi.org/10.1186/1471-2105-7-43>.

- Mathias Drton and Marloes H. Maathuis. Structure learning in graphical modeling. *Annual Review of Statistics and Its Application*, 4(1):365–393, 2017. doi: 10.1146/annurev-statistics-060116-053803. URL <https://doi.org/10.1146/annurev-statistics-060116-053803>.
- Marco Eigenmann, Sach Mukherjee, and Marloes Maathuis. Evaluation of causal structure learning algorithms via risk estimation. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pages 151–160. PMLR, 2020. URL <https://proceedings.mlr.press/v124/eigenmann20a.html>.
- Juan L. Gamella, Armeen Taeb, Christina Heinze-Deml, and Peter Bühlmann. Characterization and greedy learning of Gaussian structural causal models under unknown interventions. *arXiv preprint arXiv:2211.14897*, 2022.
- Amanda Gentzel, Dan Garant, and David Jensen. The case for evaluating causal models using interventional measures and empirical data. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/a87c11b9100c608b7f8e98cfa316ff7b-Paper.pdf.
- Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10, June 2019. doi: 10.3389/fgene.2019.00524. URL <https://doi.org/10.3389/fgene.2019.00524>.
- Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006. URL https://proceedings.neurips.cc/paper_files/paper/2006/file/e9fb2eda3d9c55a0d89c98d6c54b5b3e-Paper.pdf.
- Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu, and Bharath K. Sriperumbudur. Optimal kernel choice for large-scale two-sample tests. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper_files/paper/2012/file/dbe272bab69f8e13f14b405e038deb64-Paper.pdf.
- Patrick Healy and Nikola S. Nikolov. A branch-and-cut approach to the directed acyclic graph layering problem. In *Graph drawing*, volume 2528 of *Lecture Notes in Comput. Sci.*, pages 98–109. Springer, Berlin, 2002. ISBN 3-540-00158-1. doi: 10.1007/3-540-36151-0_{_}10. URL https://doi.org/10.1007/3-540-36151-0_10.
- Christina Heinze-Deml, Marloes H. Maathuis, and Nicolai Meinshausen. Causal structure learning. *Annual Review of Statistics and Its Application*, 5(1):371–391, 2018. doi: 10.1146/annurev-statistics-031017-100630. URL <https://doi.org/10.1146/annurev-statistics-031017-100630>.
- Patrik Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008. URL

https://proceedings.neurips.cc/paper_files/paper/2008/file/f7664060cc52bc6f3d620bcedc94a4b6-Paper.pdf.

Johannes Huegle, Christopher Hagedorn, Lukas Boehme, Mats Poerschke, Jonas Umland, and Rainer Schlosser. Manm-cs: Data generation for benchmarking causal structure learning from mixed discrete-continuous and nonlinear data. In *WHY-21 @ NeurIPS 2021*, 2021.

Aapo Hyvärinen and Stephen M. Smith. Pairwise likelihood ratios for estimation of non-Gaussian structural equation models. *J. Mach. Learn. Res.*, 14:111–152, 2013. ISSN 1532-4435,1533-7928.

A. N Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *Giornale dell’Istituto Italiano degli Attuari*, 4:83–91, 1933. URL <https://cir.nii.ac.jp/crid/1571135650766370304>.

Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural DAG learning. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=rklbKA4YDS>.

Steffen L. Lauritzen. *Graphical models*, volume 17 of *Oxford Statistical Science Series*. The Clarendon Press, Oxford University Press, New York, 1996. ISBN 0-19-852219-3. Oxford Science Publications.

Marloes Maathuis, Mathias Drton, Steffen Lauritzen, and Martin Wainwright, editors. *Handbook of graphical models*. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. CRC Press, Boca Raton, FL, 2019. ISBN 978-1-4987-8862-5.

Daniel Marbach, James C Costello, Robert Küffner, Nicole M Vega, Robert J Prill, Diogo M Camacho, Kyle R Allison, Manolis Kellis, James J Collins, and Gustavo Stolovitzky. Wisdom of crowds for robust gene network inference. *Nature Methods*, 9(8):796–804, 2012. doi: 10.1038/nmeth.2016. URL <https://doi.org/10.1038/nmeth.2016>.

Francesco Montagna, Nicoletta Noceti, Lorenzo Rosasco, Kun Zhang, and Francesco Locatello. Scalable causal discovery with score matching. In Mihaela van der Schaar, Cheng Zhang, and Dominik Janzing, editors, *Proceedings of the Second Conference on Causal Learning and Reasoning*, volume 213 of *Proceedings of Machine Learning Research*, pages 752–771. PMLR, 11–14 Apr 2023. URL <https://proceedings.mlr.press/v213/montagna23b.html>.

Joris M. Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. Distinguishing cause from effect using observational data: Methods and benchmarks. *Journal of Machine Learning Research*, 17(32):1–102, 2016. URL <http://jmlr.org/papers/v17/14-518.html>.

Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the role of sparsity and DAG constraints for learning linear DAGs. In *Advances in Neural Information Processing Systems*, volume 33, pages 17943–17954. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/d04d42cdf14579cd294e5079e0745411-Paper.pdf.

- Judea Pearl. *Causality*. Cambridge University Press, Cambridge, second edition, 2009. ISBN 978-0-521-89560-6; 0-521-77362-8. doi: 10.1017/CBO9780511803161. URL <https://doi.org/10.1017/CBO9780511803161>. Models, reasoning, and inference.
- J. Peters and P. Bühlmann. Identifiability of Gaussian structural equation models with equal error variances. *Biometrika*, 101(1):219–228, 2014. ISSN 0006-3444,1464-3510. doi: 10.1093/biomet/ast043. URL <https://doi.org/10.1093/biomet/ast043>.
- Jonas Peters and Peter Bühlmann. Structural intervention distance for evaluating causal graphs. *Neural Comput.*, 27(3):771–799, 2015. ISSN 0899-7667,1530-888X. doi: 10.1162/neco_a_00708. URL https://doi.org/10.1162/neco_a_00708.
- Jonas Peters, Joris M. Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15(58):2009–2053, 2014. URL <http://jmlr.org/papers/v15/peters14a.html>.
- Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2017. ISBN 978-0-262-03731-0. URL <https://mitpress.mit.edu/books/elements-causal-inference>.
- Pradeep Ravikumar, John Lafferty, Han Liu, and Larry Wasserman. Sparse additive models. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 71(5):1009–1030, 2009. ISSN 1369-7412,1467-9868. doi: 10.1111/j.1467-9868.2009.00718.x. URL <https://doi.org/10.1111/j.1467-9868.2009.00718.x>.
- Alexander G. Reisach, Christof Seiler, and Sebastian Weichwald. Beware of the simulated DAG! Causal discovery benchmarks may be easy to game. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 1–13. NeurIPS Proceedings, 2021.
- Alexander G. Reisach, Myriam Tami, Christof Seiler, Antoine Chambaz, and Sebastian Weichwald. Simple sorting criteria help find the causal order in additive noise models. *arXiv preprint arXiv:2303.18211*, 2023.
- Joseph M. Replogle, Reuben A. Saunders, Angela N. Pogson, Jeffrey A. Hussmann, Alexander Lenail, Alina Guna, Lauren Mascibroda, Eric J. Wagner, Karen Adelman, Gila Lithwick-Yanai, Nika Iremadze, Florian Oberstrass, Doron Lipson, Jessica L. Bonnar, Marco Jost, Thomas M. Norman, and Jonathan S. Weissman. Mapping information-rich genotype-phenotype landscapes with genome-scale perturb-seq. *Cell*, 185(14):2559–2575.e28, 2022. doi: 10.1016/j.cell.2022.05.013. URL <https://doi.org/10.1016/j.cell.2022.05.013>.
- Felix L. Rios, Giusi Moffa, and Jack Kuipers. Benchpress: A scalable and versatile workflow for benchmarking structure learning algorithms. *arXiv preprint arXiv:2107.03863*, 2022.
- Paul Rolland, Volkan Cevher, Matthäus Kleindessner, Chris Russell, Dominik Janzing, Bernhard Schölkopf, and Francesco Locatello. Score matching enables causal discovery of nonlinear additive noise models. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*,

- volume 162 of *Proceedings of Machine Learning Research*, pages 18741–18753. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/rolland22a.html>.
- Karen Sachs, Omar Perez, Dana Pe’er, Douglas A. Lauffenburger, and Garry P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721): 523–529, 2005. doi: 10.1126/science.1105809. URL <https://www.science.org/doi/abs/10.1126/science.1105809>.
- Thomas Schaffter, Daniel Marbach, and Dario Floreano. GeneNetWeaver: In silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, 27(16): 2263–2270, 2011. ISSN 1367-4803. doi: 10.1093/bioinformatics/btr373. URL <https://doi.org/10.1093/bioinformatics/btr373>.
- M. Scutari. Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010. URL <http://www.jstatsoft.org/v35/i03/>.
- Shohei Shimizu. *Statistical causal discovery: LiNGAM approach*. JSS Research Series in Statistics. Springer Japan, Tokyo, 1st ed. 2022. edition, 2022. ISBN 9784431557845.
- Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-Gaussian acyclic model for causal discovery. *J. Mach. Learn. Res.*, 7:2003–2030, 2006. ISSN 1532-4435,1533-7928.
- Shohei Shimizu, Takanori Inazumi, Yasuhiro Sogawa, Aapo Hyvärinen, Yoshinobu Kawahara, Takashi Washio, Patrik O. Hoyer, and Kenneth Bollen. DirectLiNGAM: A direct method for learning a linear non-Gaussian structural equation model. *Journal of Machine Learning Research*, 12(33):1225–1248, 2011. URL <http://jmlr.org/papers/v12/shimizulla.html>.
- Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9(1):62–72, 1991. doi: 10.1177/089443939100900106. URL <https://doi.org/10.1177/089443939100900106>.
- Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, prediction, and search*, volume 81 of *Lecture Notes in Statistics*. Springer-Verlag, New York, 1993. ISBN 0-387-97979-4. doi: 10.1007/978-1-4612-2748-9. URL <https://doi.org/10.1007/978-1-4612-2748-9>.
- Caroline Uhler, Garvesh Raskutti, Peter Bühlmann, and Bin Yu. Geometry of the faithfulness assumption in causal inference. *Ann. Statist.*, 41(2):436–463, 2013. ISSN 0090-5364,2168-8966. doi: 10.1214/12-AOS1080. URL <https://doi.org/10.1214/12-AOS1080>.
- Matthew J. Vowels, Necati Cihan Camgoz, and Richard Bowden. D’ya like DAGs? A survey on structure learning and causal discovery. *ACM Computing Surveys*, 55(4):1–36, 2022. doi: 10.1145/3527154. URL <https://doi.org/10.1145/3527154>.
- Matej Vuković and Stefan Thalmann. Causal discovery in manufacturing: A structured literature review. *Journal of Manufacturing and Materials Processing*, 6(1), 2022. ISSN 2504-4494. doi: 10.3390/jmmp6010010. URL <https://www.mdpi.com/2504-4494/6/1/10>.

- Yue Yu, Tian Gao, Naiyu Yin, and Qiang Ji. DAGs with no curl: An efficient DAG structure learning approach. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12156–12166. PMLR, 2021. URL <https://proceedings.mlr.press/v139/yu21a.html>.
- Alessio Zanga, Elif Ozkirimli, and Fabio Stella. A survey on causal discovery: Theory and practice. *International Journal of Approximate Reasoning*, 151:101–129, 2022. ISSN 0888-613X. doi: <https://doi.org/10.1016/j.ijar.2022.09.004>. URL <https://www.sciencedirect.com/science/article/pii/S0888613X22001402>.
- An Zhang, Fangfu Liu, Wenchang Ma, Zhibo Cai, Xiang Wang, and Tat-Seng Chua. Boosting causal discovery via adaptive sample reweighting. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=LNpMtk15AS4>.
- Kun Zhang and Aapo Hyvärinen. On the identifiability of the post-nonlinear causal model. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 647–655, Arlington, Virginia, USA, 2009. AUAI Press. ISBN 9780974903958.
- Zhen Zhang, Ignavier Ng, Dong Gong, Yuhang Liu, Ehsan M Abbasnejad, Mingming Gong, Kun Zhang, and Javen Qinfeng Shi. Truncated matrix power iteration for differentiable DAG learning. In *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=I4aSjFR7jOm>.
- Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/e347c51419ffb23ca3fd5050202f9c3d-Paper.pdf.
- Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric Xing. Learning sparse nonparametric DAGs. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3414–3425. PMLR, 2020. URL <https://proceedings.mlr.press/v108/zheng20a.html>.

Appendix A. Assembly line dataset

We leverage a complex production dataset comprising real measurements from a highly automated assembly line in a manufacturing plant. The assembly line consists of multiple production stations along which individual components are joined together. Parts are processed in these production stations and are sequentially passed to the next station according to the line structure.

Often, stations perform the same process several times either in succession or in parallel to reduce overall cycle time. Raw components are added to the assembly at the respective stations. The quality of the assembly process depends on the respective machine state and settings (e.g. tool state, control parameters), the environmental conditions, and on the raw state of the components or intermediates. We carefully study the physical processes that take place in each of the stations.

A.1. Mapping process knowledge

Press-in In order for the inner cylindrical component to be pressed into the bore, mechanical force has to exceed friction force. The friction force depends on component properties, such as the geometrical dimensions (especially diameters and axial lengths), structural stiffness as well as the surface conditions (e.g. surface roughness). Also, machine properties such as misalignment of the pressing machine tool with respect to the work piece holder and cylindrical bore axis can affect the process. None of these properties is measured directly for each component. Depending on the product type, the nominal values of the component properties may differ.

During the fitting process the force acting on the pressing machine tool and its current vertical position are measured jointly using special sensors. In order to control and monitor the process both signals are analyzed during and after the process by the process control unit, for example using gradient detection algorithms. The change in measured position is the sum of both the elastic deformation of the pressing machine (driven by the force and structural machine stiffness) and the actual movement of the inner component relative to the bore. Figure 8 demonstrates how the machine tool force behaves along the vertical position. The most important features extracted from this procedure are stored in the manufacturing execution system. Table 1 depicts a selection of these and lays out dependencies.

Staking In staking the deformation process of the material is mainly driven by two factors. One factor is the material yield strength and hardening behavior. The other factor is the compression stress. The latter is imposed by the force acting on the tool and by the contact area between tool and component. If several staking processes are happening on the same component with several bore positions, the unknown material state is likely to be similar, confounding the observed force values. Again a typical force and position curve of a staking process as well as a description of extracted features including their interrelations are demonstrated in Figure 9 and Table 2, respectively.

Often, press-in and staking processes are run on the same machine. Unobserved machine properties such as structural stiffness, tool misalignment and force sensor positions may impact both processes. Furthermore, confounding can be caused by similarities of the component properties (same component with several bores) or due to batch effects.

Figure 10 depicts an exemplary press-in and staking process. Depending on the production cell and process step these might vary slightly in size and complexity along the production line. From Figure 10 we can see that the staking process is larger and more complex than the press-in process. This reflects the fact that a staking process is an extension of a press-in process.

Table 1: Description of relevant force and position features extracted from the measurements during the press-in process.

Parameter	Description
F_{offs}	Average force measured during closing of the gap between tool and work piece. This depends on the acceleration and mass of the tool as well as the force sensor measurement error.
F_1	Press-in force (friction force) required to reach the block position (S_1) as extracted from gradient detection algorithm.
F_2	Maximum force at the end of the process. This force depends on the structural stiffness and the difference between the positions S_2 and S_1 .
S_0	Axial position of tool at first force increase as extracted from gradient detection algorithm. Position depends on geometrical tolerances of the machine and components as well as on the calibration of the position sensor.
S_1	Axial position of tool at reaching the block position. Position depends on the relative displacement S_0 . S_1 is affected by the length of the bore as well as by the elastic deformation of the machine. Deformation is influenced by the required press-in force F_1 and by the structural machine and component stiffness. Once this gradient position is detected by the process control unit during process execution the command for reversing the tool movement is given.
S_2	Maximum tool position where the tool direction of movement is effectively reversed. This is affected by the position S_1 , by the time required for computation of the gradient point, the tool speed, by the inertial mass of the tool and by the effective structural stiffness.

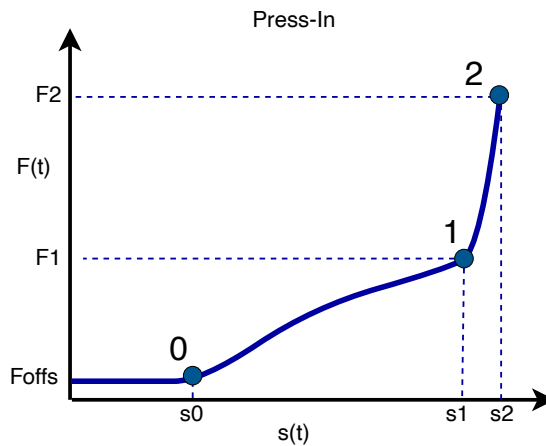


Figure 8: A typical Force (y-axis) and Position (x-axis) curve during press-in illustrating the characteristics of the process.

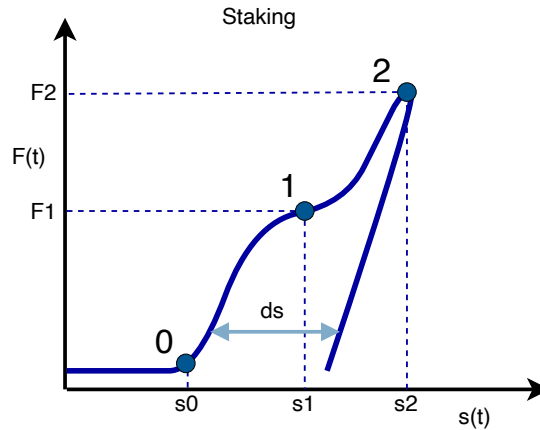


Figure 9: A typical Force (y-axis) and Position (x-axis) curve during staking illustrating the characteristics of the process.

Table 2: Description of relevant force and position features extracted from the measurements during the staking process.

Parameter	Description
F_1^*	Staking force required to reach the turning point of increasing structural stiffness as extracted from the detection algorithm. Once this turning point is detected by the process control unit during process execution the process command for reversing the tool movement is given after an additional, pre-defined force delta has been exceeded.
F_2^*	Maximum force at the end of the process. This force depends on the structural stiffness and the difference between the positions S_2^* and S_1^* .
S_0^*	Axial position of tool at first force increase as extracted from gradient detection algorithm. Position depends on geometrical tolerances of the machine and components as well as on the calibration of the position sensor.
S_1^*	Axial position of tool at reaching the turning point. Position depends on the relative displacement S_0 . S_0 is affected by the length of the bore as well as by the elastic deformation of the machine. Deformation is influenced by the required press-in force F_1^* and by the structural machine and component stiffness.
S_2^*	Maximum tool position where the tool direction of movement is effectively reversed. This is affected by the position S_1^* , by the time required for computation of the turning point, the tool speed, by the inertial mass of the tool and by the effective structural stiffness.
ΔS^*	Difference in axial position of the tool after removing the pressing force measured at a pre-defined remaining force value. This indicates the true deformation length in the bore without the elastic machine deformation.

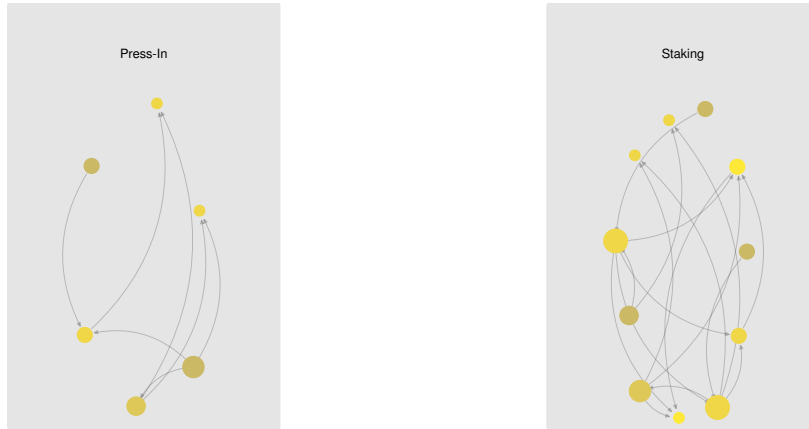


Figure 10: Process DAGs of an exemplary press-in and staking process. Node size increases with the number of out-edges. Node color gets brighter with the number of in-edges.

Appendix B. Proof of Proposition 1

Proof It suffices to show $\Pi_L \subseteq \Pi_{L'}$. So, let $\pi \in \Pi_L$ and consider any two nodes $v, w \in V$. If $v \in V_s$ and $w \in V_t$ with $s \neq t$, then we must have $s < t$ because L is a layered DAG. Since L' is based on the same partition \mathcal{L} , we have $\pi \in \Pi_{L'}$. If instead $v, w \in V_s$ for some $s \in [K]$, then $L_s = L'_s$. We conclude that v is a descendant of w in L if only if this is the case in L' . Thus, again $\pi \in \Pi_{L'}$. ■

Appendix C. Semisynthetic data generator

The semisynthetic data generation tool is available through the Python library `causalAssembly` (<https://github.com/boschresearch/causalAssembly>). In the library, we also provide the semisynthetic data itself (and for convenience one set of datasets with sample size $n = 500$) as well as the partial ground truth. All benchmark results presented both in the main text and the supplementary sections will also be accessible through the library.

C.1. Quality of the DRF procedure

To avoid overfitting when learning DRFs, we grow $N = 2000$ trees on a random subset of the training data for each conditional distribution. When forming each tree, the subsets are split again; one part is used to grow the tree, the other part is used to populate the leaf nodes. We choose the Gaussian kernel for the MMD splitting criterion with bandwidth set according to the *median heuristic* (Gretton et al., 2012). Table 5 in Section D.3 gives an overview over the all relevant hyperparameters.

In the interest of quality control, we compare correlation patterns in the synthetic data largely agree with those of the real data as shown in Figure 11. Some divergence is to be expected as the

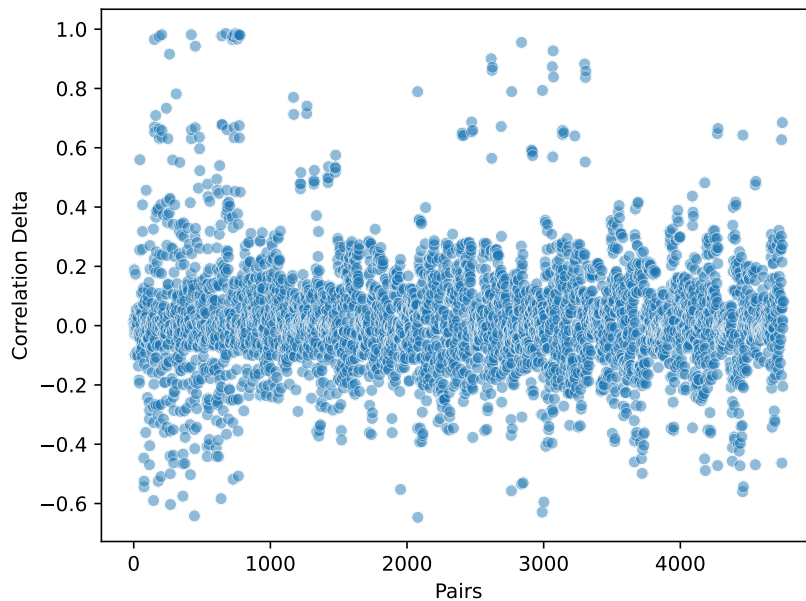


Figure 11: Pairwise correlation agreement between semisynthetic and real data. We depict the difference between the pearson correlation of the same pair in one draw of the semisynthetic data and the real production line data. Sample size is $n = 15.581$ for semisynthetic and real data.

real data comes with a wide range of confounding problems which our procedure deals with by employing the DRF procedure described in Section 3.4. Figure 11 confirms that for most of the pairs among the $p = 98$ variables, correlation patterns agree.

C.2. Influence of initial learning

Before investigating the influence of the absence of prior knowledge, we engage in a brief comparison of between and within process edge recovery. Figure 12 demonstrates that cross-process edges are indeed not recovered more easily as opposed to within-process edges. If any, the opposite is true and within-process edges achieve somewhat better precision and recall scores.

Turning to the absence of prior knowledge, learning edges within and across processes using causal discovery algorithms only without prior knowledge leaves distinct traces. Benchmarking outcomes are then tilted in favor of the routine employed in this initial step. We demonstrate this on processes that take place in Station 3 consisting of 16 nodes. We ignore existing process knowledge and learn from the data only. Figures 13 and 14 report SHD and F_1 score between the ground truth DAG learned by the initial causal discovery procedure and results based on corresponding semisynthetic data.

Next to overall patterns left by the initial learning step, we present sensitivity results with respect to measurement scale. The real data obtained from the process control unit carry a wide range of different measurement scales. These scales should ideally be irrelevant with respect to causal relationships. In a first step, we keep the raw data scale both for the initial learning and for the

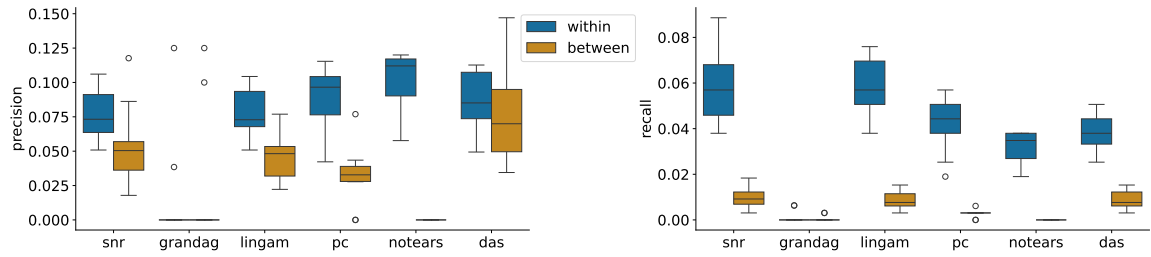


Figure 12: Comparison between within and cross process edges. Simulation run with 50 repetitions and sample size $n = 5000$. Precision and recall scores are relative to the amount of within and cross-process edges, respectively.

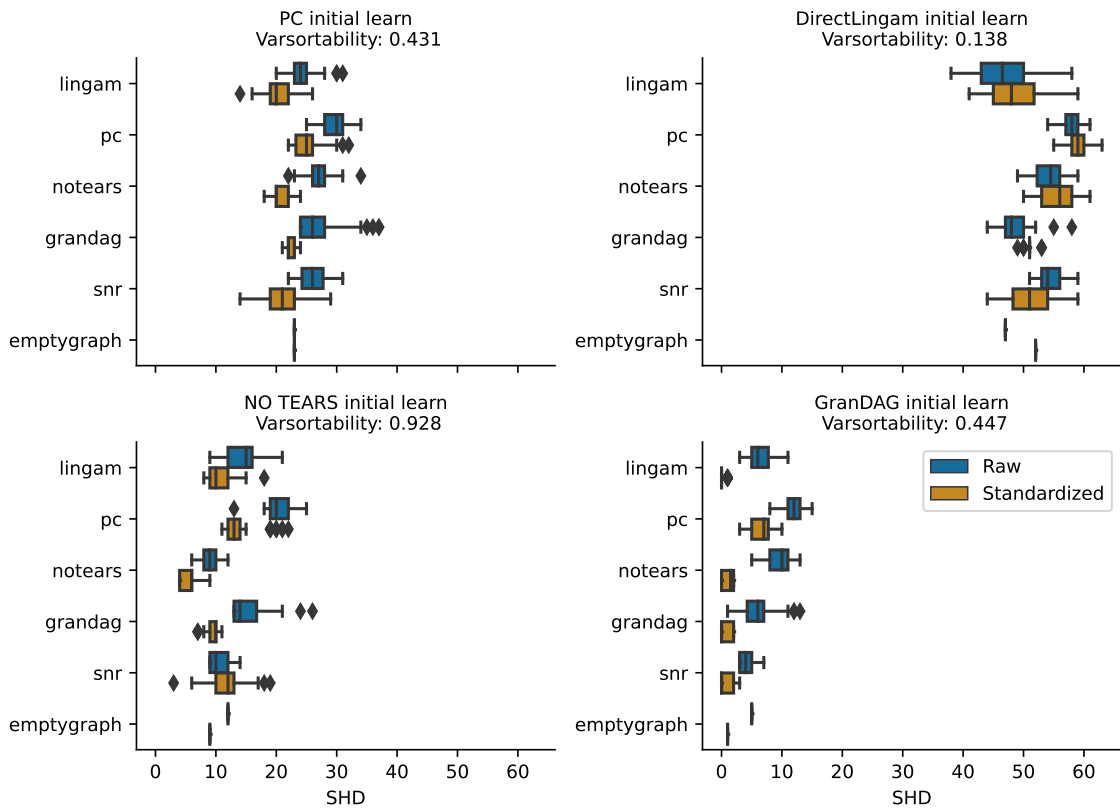


Figure 13: SHD for initial learning based on chosen causal discovery algorithms. Simulation run with 50 repetitions.

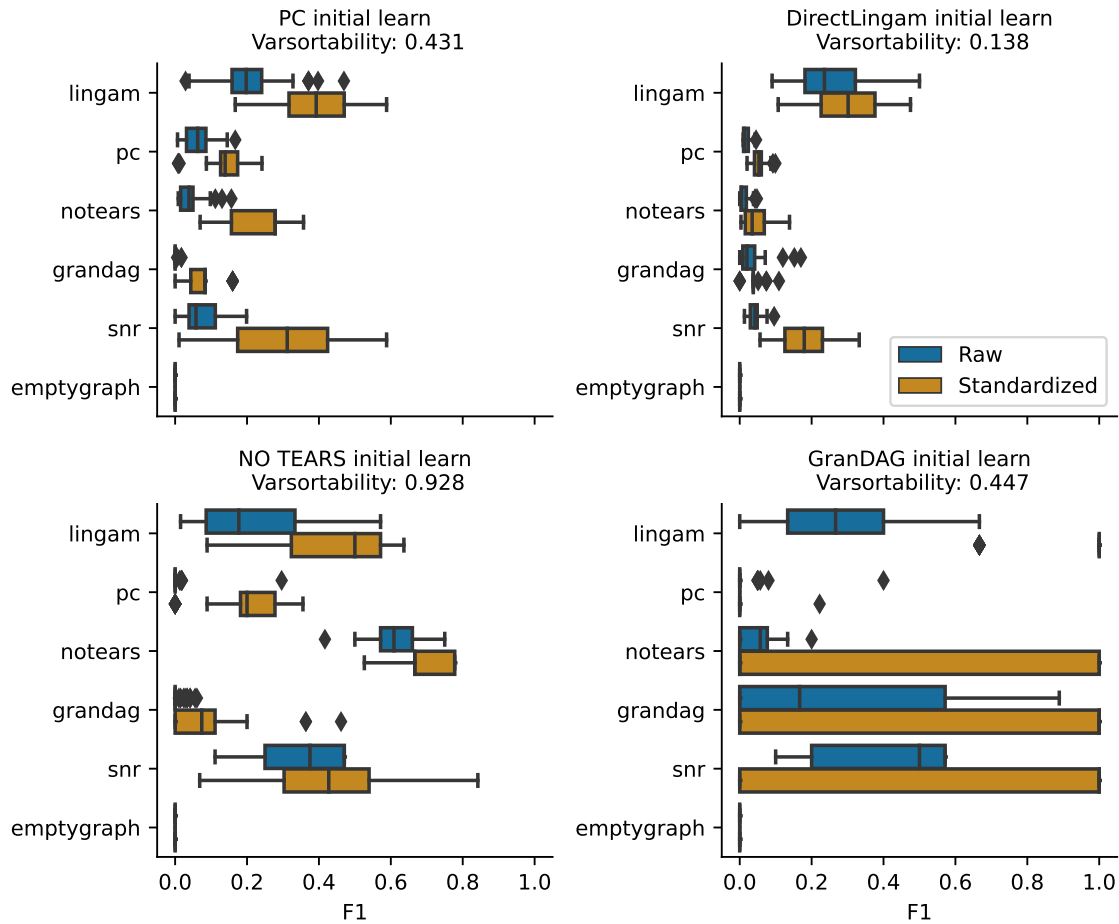


Figure 14: F_1 score for initial learning based on chosen causal discovery algorithms. Simulation run with 50 repetitions.

benchmark run based on the obtained ground truth. We report the average varsortability across simulation runs. In a second step, we standardize all data. We also report results for the empty graph to keep track of the number of edges inferred during the initial graph learning step. If the SHD of the empty graph changes under standardization then data scale has an effect on the initial ground truth.

DirectLiNGAM, NOTEARS, and GraN-DAG show a clear performance edge when the ground truth is originally obtained using the respective routine. This pattern is most clear in Figure 14. The PC algorithm does not seem to benefit too greatly which can to some degree be attributed to the random DAG completion. In contrast to the remaining methods, the PC algorithm is agnostic to measurement scale. DirectLiNGAM is in theory also scale agnostic, however scaling may influence the adjacency matrix estimation via adaptive lasso when predictors are not standardized. Regarding NOTEARS and GraN-DAG measurement units play a substantial role. NOTEARS obtains a ground truth with varsortability close to one on unstandardized data and recovers a very sparse DAG on standardized data. This suggests that NOTEARS essentially orders variables according to their variances when learned on raw data. GraN-DAG results are highly variable throughout all initial learning methods. The instability becomes particularly severe when applied to standardized data.

Overall, except for the PC algorithm there are clear traces left by initial learning algorithms applied to process cell structure learning discovery. In this context, the question arises whether to simply select a random DAG to represent process or station relationships. Following the proposed procedure and given sufficient sample size, the DRFs will be able to produce weights that allow drawing from such a ground truth graph. The choice of random DAG however influences strongly how close resulting synthetic data to the real world complexity will be. In the complete absence of ground truth knowledge, such a procedure might be an alternative to specifying a random SCM.

Appendix D. Benchmarking

D.1. Algorithms and metrics

Algorithms The following well-known structure learning algorithms are employed in our work.

PC algorithm (Spirites et al., 1993) (`pc`)

A constraint based algorithm that in a first step performs conditional independence tests in a resource efficient way returning a skeleton and in a second step orients as many edges as possible. We use the stable version of the algorithm proposed by Colombo and Maathuis (2014). Conditional independence tests are performed using Fisher z-transformed (partial) Pearson correlation coefficients. Since the PC algorithm outputs a completed partially directed acyclic graph (CPDAG), we choose in each run a random DAG that is a member of the MEC implied by the CPDAG.

DirectLiNGAM (Shimizu et al., 2011; Hyvärinen and Smith, 2013) (`lingam`)

The direct method to learning the LiNGAM model (Shimizu et al., 2006) referred to in Section 2.3. First, the procedure detects source variables by employing entropy-based measures to find independence between the error variables. The *most independent* variable is placed first in the causal ordering and its effect removed from the other variables. As the ordering of the residuals resembles that of the variables themselves the algorithm continues to find the next variable in the ordering by finding the next *source*-residual.

NOTEARS (Zheng et al., 2018, 2020) (`notears`)

A score-based method that recasts the discrete DAG constraint as a continuous equality constraint. This allows for the application of a wide range of techniques for non-convex programs. In NOTEARS a linear SCM is assumed and for each variable, a feed-forward neural network is trained that gets all

other variables as input. The graph structure is then dynamically read-off from the networks during training by aggregating the weights of the input layer of each network. Acyclicity of the graph is enforced by training the networks simultaneously in an augmented Lagrangian schema such that the matrix exponential of the adjacency matrix has small trace.

GraN-DAG (Lachapelle et al., 2020) (`grandag`)

A score-based method and an extension of NOTEARS. In contrast to NOTEARS, the relation structure is read-off from the networks by aggregating all the weights till the output layer of the neural networks individually.

SCORE and *DAS* (Rolland et al., 2022; Montagna et al., 2023) (`score`, `das`)

An order-based method for learning nonlinear additive Gaussian noise models which has been shown to be identifiable from observational data (Peters et al., 2014). The method evolves around estimating the causal ordering based in approximating the score (the map $\nabla \log p(x)$ for a differentiable density $p(x)$) of the data distribution. Under this model, leaf nodes indicate a constant entry in the associated diagonal element of the score’s Jacobian. This allows for sequential leaf node identification implying a causal ordering. The method concludes with a pruning step to remove spurious edges. Its extension *discovery at scale* (DAS) reduces the complexity of the pruning step.

sortnregress (Reisach et al., 2021) (`snr`)

A diagnostic tool to expose data scenarios with high varsortability. The procedure sorts variables by increasing marginal variance and then regresses each variable on their ancestors adding an ℓ_1 type penalty to induce sparsity.

Metrics The choice of evaluation metrics for assessing causal discovery outcomes matters (see Gentzel et al., 2019). We employ a selection of common structural performance metrics and note that in a full scale benchmark these should be extended to include a greater variety of evaluation measures.

Precision, Recall and F_1 score:

Precision in the graphical context refers to the fraction of correctly identified edges among all edges inferred. Recall, sets these correctly identified edges in relation to all edges present in the ground truth. The F_1 score is the harmonic mean of Precision and Recall, i.e. $F_1 = 2 \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$.

Structural hamming distance (SHD):

The SHD counts how many edge insertions, deletions, and reversals are necessary to turn the estimated DAG into the ground truth DAG.

Structural Interventional Distance (SID) (Peters and Bühlmann, 2015):

The SID counts the number of incorrectly inferred interventional distributions in the estimated DAG when compared to the ground truth DAG.

Harmonization We employ two harmonization strategies *CPDAG-transform* and *average-dag*.

(i) We transform all graphs, including the ground truth DAG, to their completed partially directed acyclic graph (CPDAG) representing the associated MEC (see Andersson et al., 1997, for details). Metrics will be calculated on CPDAGs only. Note that the SID is defined for DAGs and will not be part of the evaluation metrics following *CPDAG-transform*.

(ii) The second strategy, *average-dag*, involves choosing one member DAG from the implied MEC of algorithms that do not output a DAG. We then enumerate all DAGs in the MEC, calculate their structural Hamming distance (SHD) with respect to the ground truth DAG, and take the one whose SHD is closest to the average SHD. When running experiments on the full line, enumerating

Table 3: All benchmarks are run on a Linux machine with 32 GB Memory and 8 cores.

Result	Compute time
Full assembly line benchmark Fig 6 and 19 in the main text	16.90 hours
Station 1 benchmark Fig 15 and 20	1.07 hours
Station 2 benchmark Fig 16 and 21	4.39 hours
Station 3 benchmark Fig 7 and 22	2.22 hours
Station 4 benchmark Fig 17 and 23	3.54 hours
Station 5 benchmark Fig 18 and 24	2.33 hours

Table 4: Average varsortability over 100 draws using the semisynthetic data generation procedure for the full assembly line and for each production station.

	Varsortability
Full line	0.486122
Station1	0.669167
Station2	0.348319
Station3	0.330588
Station4	0.247204
Station5	0.210417

DAGs in their MEC becomes computationally infeasible, so we choose one random DAG from the associated MEC.

D.2. Additional benchmarking results

We present additional benchmarks for the CPDAG harmonization strategy and the remaining production stations from the MEC enumeration strategy. Average varsortability before standardization is reported in Table 4 and Table 3 states compute time for each of the benchmark scenarios. As expected, varsortability ranges between 0.7 in the first station and 0.22 in station five. The full assembly line exhibits varsortability close to 0.5 indicating that ordering variables by their marginal variances does not give any insights regarding the causal ordering. In each simulation run, we sample from the DRF regrown on the corresponding layer-induced subgraphs.

We find that results vary substantially across stations. Figure 15 and 20 report results for the first station. DirectLiNGAM performs best while NOTEARS is not able to detect any edges. GraN-DAG results are very unstable throughout all benchmarks. The PC algorithm performs only slightly worse than DirectLiNGAM and SCORE’s performance dropped when transformed to the corresponding MEC.

In Figure 16 and 21 Station 2 two benchmarks are depicted. In this case, NOTEARS performs best in the context of recovering the true DAG. When converted to CPDAGs, NOTEARS performance drops and PC algorithm performance increases drastically. Figure 22 reports Station 3 benchmarks for the CPDAG harmonization strategy. In this case the PC algorithms outperforms all other routines when compared to the ground truth CPDAG. In Figures 17 and 23 Station 4 benchmarks suggest again that the PC algorithm outperforms all other causal discovery algorithms. On station five (Figure

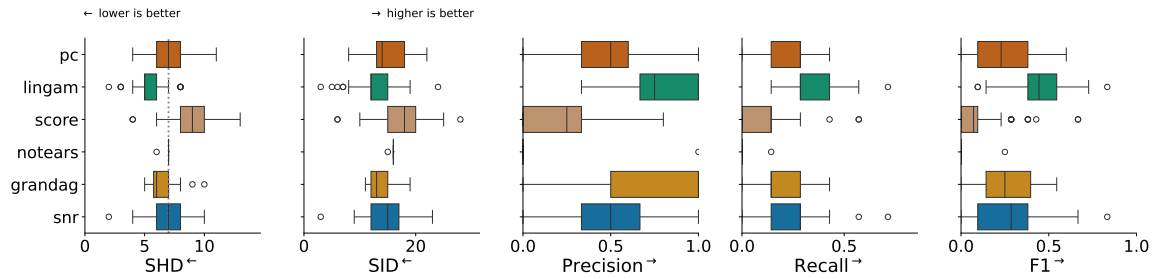


Figure 15: Chosen metrics for assessing causal discovery outcomes on Station 1 ($|V_1 \cup V_2| = 6$) using 100 simulation runs on a sample size of $n = 500$.

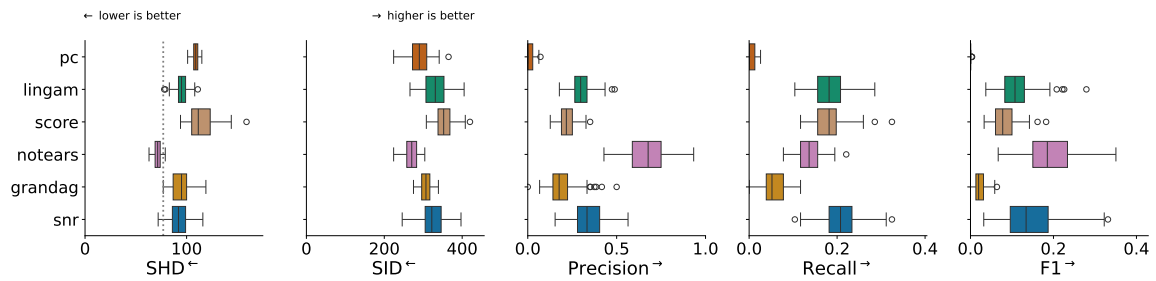


Figure 16: Chosen metrics for assessing causal discovery outcomes on Station 2 ($|V_3 \cup V_4| = 34$) using 100 simulation runs on a sample size of $n = 500$.

18 and 24) SCORE overall performs best on DAG level and the PC algorithm draws equal for the CPDAG harmonization strategy. Last, Figure 19 illustrates the full line benchmark results for the CPDAG harmonization strategy. As mentioned in the main text, NOTEARS performs best overall, GraN-DAGs behavior is very unreliable, and the PC algorithm is the runner-up. Yet, none of the procedures is able to perform significantly better than *random regress*.

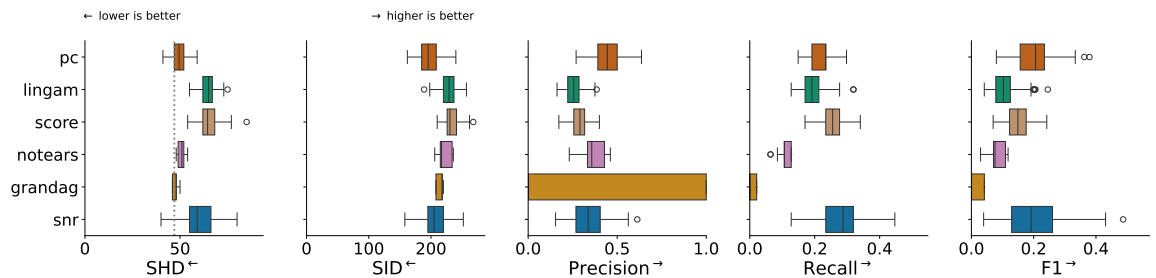


Figure 17: Chosen metrics for assessing causal discovery outcomes on Station 4 ($|V_7 \cup V_8| = 26$) using 100 simulation runs on a sample size of $n = 500$.

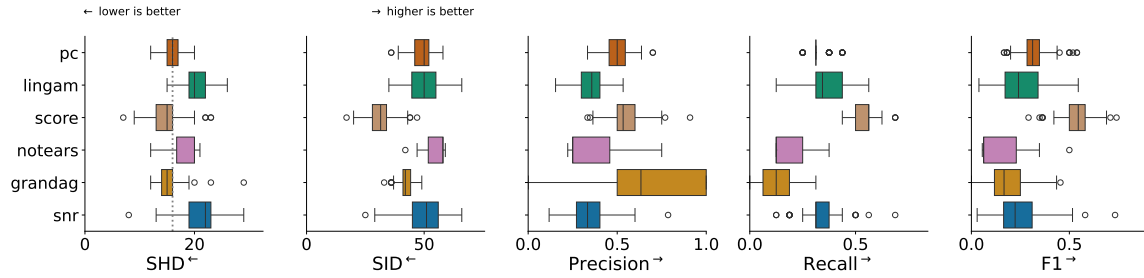


Figure 18: Chosen metrics for assessing causal discovery outcomes on Station 5 ($|V_9 \cup V_{10}| = 16$) using 100 simulation runs on a sample size of $n = 500$.

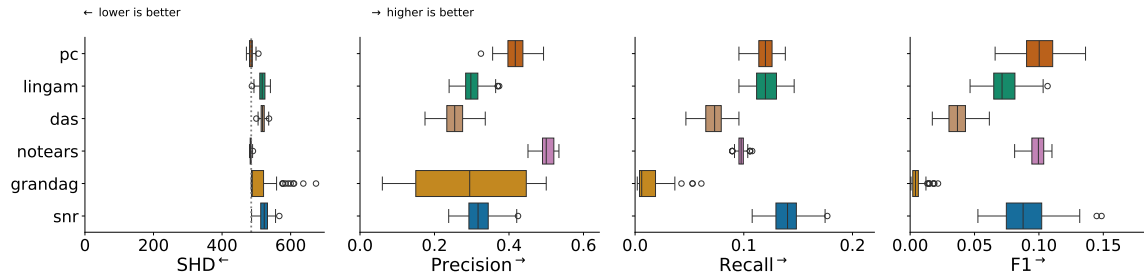


Figure 19: Chosen metrics for assessing causal discovery of the full assembly line based on CPDAG comparisons ($|V| = 98$) using 100 simulation runs on a sample size of $n = 5000$. The dashed line corresponds to the SHD between the empty graph and ground truth indicating the number of edges in the ground truth CPDAG.

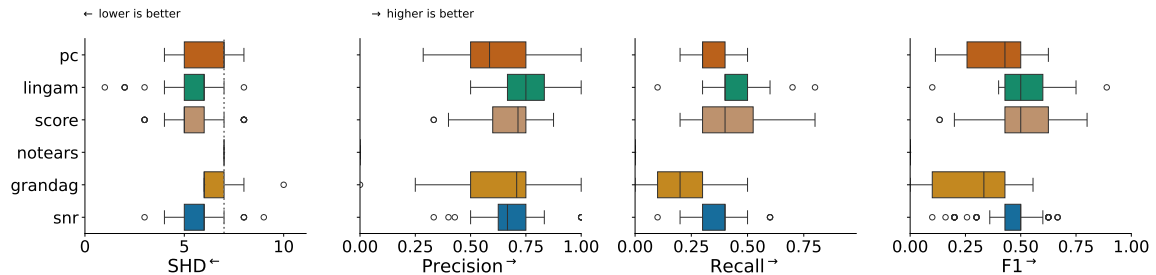


Figure 20: Chosen metrics for assessing causal discovery outcomes on Station 1 ($|V_1 \cup V_2| = 6$) after converting all DAGs to CPDAGs using 100 simulation runs on a sample size of $n = 500$.

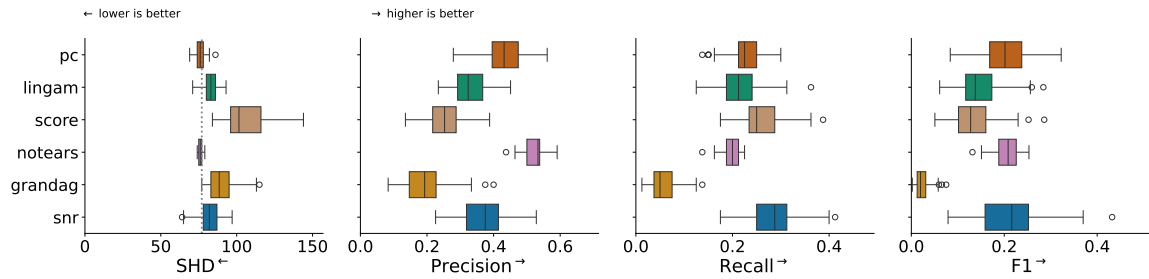


Figure 21: Chosen metrics for assessing causal discovery outcomes on Station 2 ($|V_3 \cup V_4| = 34$) after converting all DAGs to CPDAGs using 100 simulation runs on a sample size of $n = 500$.

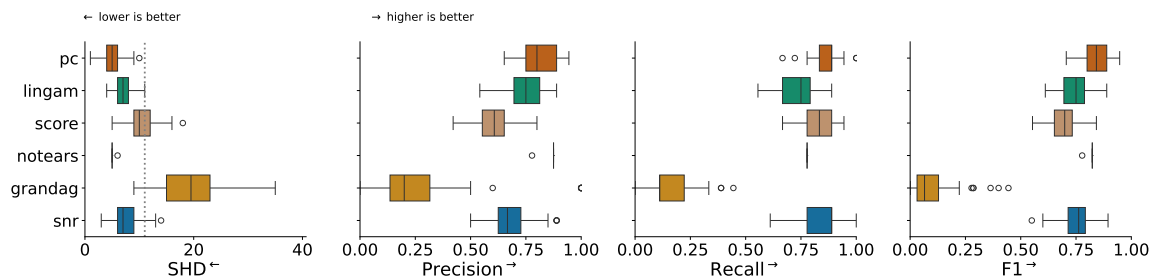


Figure 22: Chosen metrics for assessing causal discovery outcomes on Station 3 ($|V_5 \cup V_6| = 16$) after converting all DAGs to CPDAGs using 100 simulation runs on a sample size of $n = 500$.

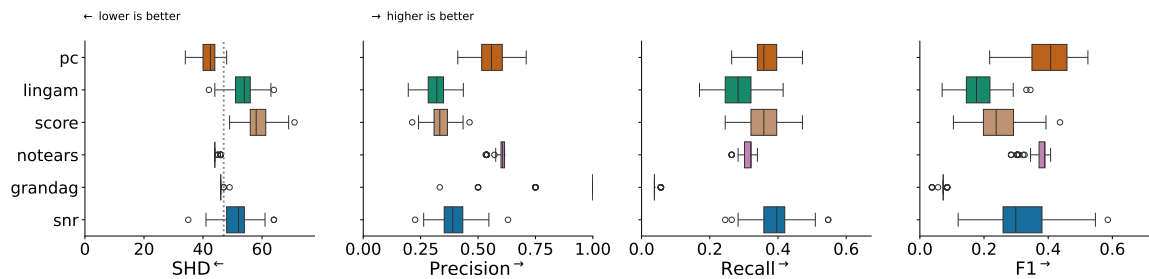


Figure 23: Chosen metrics for assessing causal discovery outcomes on Station 4 ($|V_7 \cup V_8| = 26$) after converting all DAGs to CPDAGs using 100 simulation runs on a sample size of $n = 500$.

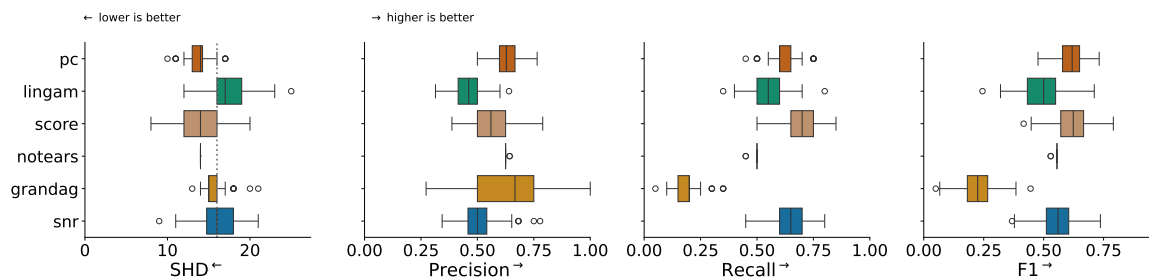


Figure 24: Chosen metrics for assessing causal discovery outcomes on Station 5 ($|V_9 \cup V_{10}| = 16$) after converting all DAGs to CPDAGs using 100 simulation runs on a sample size of $n = 500$.

D.3. Hyper and tuning parameters

Table 5: Hyper and tuning parameter choices for routines employed in this work.

	Parameters
SpAM	<pre> num_folds = 5 cv_loss = MSE lambda_choice = 1se num_bsplines = 6 </pre>
DRF	<pre> min_node_size = 15 num_trees = 2000 splitting_rule = FourierMMD </pre>
PC Algorithm	<pre> indep_test : fisherz alpha = 0.05 </pre>
NOTEARS	<pre> lambda_1 = 0.1 loss_type = ℓ_2 max_iter = 100 h_tol = $1e - 8$ rho_max = $1e + 16$ w_threshold = 0.3 </pre>
GraN-DAG	<pre> input_dim = V hidden_num = 2 hidden_dim = 10 batch_size = 64 lr = 0.001 iterations = 10000 model_name = NonLinGaussANM nonlinear = leaky-relu optimizer = rmsprop h_threshold = $1e - 8$ lambda_init = 0.0 mu_init = 0.001 omega_lambda = 0.0001 omega_mu = 0.9 stop_crit_win = 100 </pre>

edge_clamp_range = 0.0001

SCORE, DAS

eta_G = 0.001

eta_H = 0.001

alpha = 0.05

n_splines = 10

splines_degree = 3

Table 5 reports all hyperparameters and tuning parameter choices for benchmark results as well as for SpAM and DRF fitting. All benchmark algorithms were taken off-the-shelf to avoid providing any of the procedures with an unfair advantage.