Sensitivity Analysis of Linear Structural Causal Models

Carlos Cinelli [1] **Daniel Kumor** [2] **Bryant Chen** [3] **Judea Pearl** [1] **Elias Bareinboim** [2]

Abstract

Causal inference requires assumptions about the data generating process, many of which are unverifiable from the data. Given that some causal assumptions might be uncertain or disputed, formal methods are needed to quantify how sensitive research conclusions are to violations of those assumptions. Although an extensive literature exists on the topic, most results are limited to specific model structures, while a general-purpose algorithmic framework for sensitivity analysis is still lacking. In this paper, we develop a formal, systematic approach to sensitivity analysis for *arbitrary* linear Structural Causal Models (SCMs). We start by formalizing sensitivity analysis as a constrained identification problem. We then develop an efficient, graph-based identification algorithm that exploits non-zero constraints on both directed and bidirected edges. This allows researchers to systematically derive sensitivity curves for a target causal quantity with an arbitrary set of path coefficients and error covariances as sensitivity parameters. These results can be used to display the degree to which violations of causal assumptions affect the target quantity of interest, and to judge, on scientific grounds, whether problematic degrees of violations are plausible.

1. Introduction

Randomized controlled trials (RCT) are considered the gold standard for identifying cause-effect relationships in data-intensive sciences (Giffin et al., 2010). In practice, however, direct randomization is often infeasible or unethical, requiring researchers to combine non-experimental observations with assumptions about the data generating process in order to obtain causal claims. These assumptions are usually encoded as the absence of certain causal relationships, or as the absence of association between certain unobserved factors. Conclusions based on causal models are, therefore, provisional: they depend on the validity of causal assumptions, regardless of the sample size (Pearl, 2000; Spirtes et al., 2000; Bareinboim & Pearl, 2016).

In many real settings, it is not uncommon that these assumptions are subject to uncertainty or dispute. Scientists may posit alternative causal models that are equally compatible with the observed data; or, more mundanely, researchers can make identification assumptions for convenience, simply to proceed with estimation.[1] Regardless of the motivation, the provisional character of causal inference behooves us to formally assess the extent to which causal conclusions are *sensitive* to violations of those assumptions.

The importance of such exercises is best illustrated with a real example, which directly impacted public policy. During the late 1950s and early 1960s, there was a fierce debate regarding the causal effect of cigarette smoking on lung cancer. One of its most notable skeptics was the influential statistician Ronald Fisher, who claimed that, without an experiment, one cannot rule out unobserved common causes (e.g. the individual's genotype) as being responsible for the observed association (Fisher, 1957; 1958). Technically speaking, Fisher's statement was accurate; data alone could not refute his hypothesis. Yet, although no RCT measuring the effect of cigarette smoking on lung cancer was performed, currently there exists a broad consensus around the issue. How could such a consensus emerge?

An important step towards the current state of affairs was a *sensitivity analysis* performed by Cornfield et al. (1959). Their investigation consisted of the following *hypothetical* question: if Fisher's hypothesis were true, *how strong* would the alleged confounder need to be *to explain all* the observed association between cigarette smoking and lung cancer? The analysis concluded that, since smokers had nine times the risk of nonsmokers for developing lung cancer, the latent confounder would need to be at least nine times

[1]Depts. of Statistics and Computer Science, University of California, Los Angeles, California, USA. [2]Dept. of Computer Science, Purdue University, West Lafayette, IN, USA. [3]Brex, San Francisco, CA, USA. Most of this work was conducted while at IBM Research AI. Correspondence to: Carlos Cinelli <carloscinelli@ucla.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

[1]As noted by Joffe et al. (2010), "such assumptions are usually made casually, largely because they justify the use of available statistical methods and not because they are truly believed".

more common in smokers than in nonsmokers—something deemed implausible by experts at the time.

Cornfield's exercise reveals the fundamental steps of a sensitivity analysis. The analyst introduces a *violation* of a causal assumption of the current model, such as positing the presence of unobserved confounders that induce a non-zero association between two error terms. Crucially, however, we are willing to tolerate this violation up to a certain *plausibility limit* dictated by expert judgment (e.g., prior biological understanding, pilot studies). The task is, thus, to systematically quantify how different hypothetical "degrees" of violation (to be defined) affect the conclusions, and to judge whether expert knowledge can rule out problematic values.

The problem of sensitivity analysis has been studied throughout the sciences, ranging from statistics (Rosenbaum & Rubin, 1983; Small, 2007; Rosenbaum, 2010; Cinelli & Hazlett, 2018; Franks et al., 2019) to epidemiology (Brumback et al., 2004; Vanderweele & Arah, 2011; Ding & VanderWeele, 2016; Arah, 2017), sociology (Frank, 2000), psychology (Mauro, 1990), political science (Imai et al., 2010; Blackwell, 2013), and economics (Leamer, 1983; Imbens, 2003; Oster, 2017; Masten & Poirier, 2018). Notwithstanding all this attention, the current literature is still limited to specific models and solved on a case-by-case basis. Considering the ubiquity of causal questions in the sciences and artificial intelligence, a formal, algorithmic framework to deal with violations of causal assumptions is needed.

Causal modeling requires a formal language where the characterization of the data generating process can be encoded explicitly. Structural Causal Models (Pearl, 2000) provide such a language and, in many fields, including machine learning, the health and social sciences, linearity is a popular modeling choice. In this paper, we focus on the sensitivity analysis of linear acyclic semi-Markovian SCMs. We allow violations of exclusion and independence restrictions, such as (i) the absence or presence of unobserved common causes; and, (ii) the absence, presence or reversal of direct causal effects. Our contributions are the following:

1. We introduce a formal, algorithmic approach for sensitivity analysis in linear SCMs and show it can be reduced to a problem of *identification with non-zero constraints*, i.e, identification when certain parameter values are fixed to a known, but non-zero, number.
2. We develop a novel graphical procedure, called PUSH-FORWARD, that reduces identification with a known error covariance to vanilla identification, for which a plethora of algorithms are available.
3. We develop an efficient graph-based constrained identification algorithm that takes as input a set of sensitivity parameters and returns a sensitivity curve for the effect estimate. The algorithm is theoretically sound and experimental results corroborate its generality, showing canonical sensitivity analysis examples are a small subset of the cases solved by our proposal.

This paper is structured as follows. Section 2 reviews basic terminology and definitions that will be used throughout the text. Section 3 shows how sensitivity analysis in the context of linear SCMs can be reduced to a constrained identification problem. In Section 4 we develop a novel approach that allows researchers to systematically incorporate constraints on error covariances of linear SCMs. Section 5 utilizes these results to construct a constrained identification algorithm for deriving sensitivity curves. Finally, Section 6 presents experimental results to evaluate our proposals.

2. Preliminaries

In this paper, we use the language of structural causal models as our basic semantic framework (Pearl, 2000). In particular, we consider linear semi-Markovian SCMs, consisting of a set of equations of the form $V = \Lambda V + U$, where V represent the endogenous variables, U the exogenous variables, and Λ a matrix containing the *structural coefficients* representing both the strength of causal relationships and lack of direct causation among variables (when $\lambda_{ij} = 0$). The exogenous variables are usually assumed to be multivariate Gaussian with covariance matrix $\mathcal{E}$, encoding independence between error terms (when $\epsilon_{ij} = 0$).[2] We focus on acyclic models, where Λ can be arranged to be lower triangular.

The covariance matrix Σ of the endogenous variables induced by model M is given by $\Sigma = (I - \Lambda)^{-1}\mathcal{E}(I - \Lambda)^{-\top}$. Without loss of generality, we assume model variables have been standardized to unit variance. For any three variables x, y and z, we denote σ_{yx} to be the covariance of x and y, $\sigma_{yx.z}$ to be the partial covariance of y and x given z, and $R_{yx.z}$ the regression coefficient of y on x adjusting for z. Causal quantities of interest in a linear SCM are usually entries of Λ (or functions of those entries), and identifiability reduces to checking whether they can be uniquely computed from the observed covariance matrix Σ.

Causal graphs provide a parsimonious encoding of some of the substantive assumptions of a linear SCM. The causal graph (or the path diagram) of model M is a graph $G = (V, D, B)$, where V denotes the vertices (endogenous variables), D the set of directed edges (non-zero entries of Λ) and B the set of bidirected edges (non-zero entries of $\mathcal{E}$). Missing directed edges represent *exclusion restrictions*—a variable is not a direct cause of the other. Missing bidirected edges denote *independence restrictions*, representing the fact that no latent common causes exist between two observed variables. When clear from context, we may

[2] Gaussianity is not necessary for the results of the paper.

treat model coefficients and their corresponding edges on the graph interchangeably. We use standard graph notation, where $Pa(y)$ denotes the parents, $Ch(y)$ the children, $Anc(y)$ the ancestors, and $De(y)$ the descendants of node y.

3. Sensitivity analysis and identification

In this section we demonstrate the pervasiveness of identification problems in sensitivity analysis in the context of a simple example. Suppose a scientist hypothesizes model G_O shown in Fig. 1a with the goal of estimating the direct effect of a treatment x on an outcome y (structural coefficient λ_{xy}). By the single-door criterion (Pearl 2000), she verifies λ_{xy} is identifiable in G_O and equal to the regression estimand $R_{yx.z}$, licensing her to proceed with estimation.

Another investigator, however, is suspicious of the bold assumption that no common causes (confounders) exist between z and x in G_O. She goes on, therefore, and constructs an alternative model G_A (Fig. 1b) such that the bidirected edge $z \leftrightarrow x$ is included to account for that possibility. A question now naturally arises: how wrong could one be using $R_{yx.z}$ to estimate λ_{xy} if the true causal model were given by graph G_A? Answering this question requires defining a measure of "wrongness" of the estimand, and perhaps the simplest such measure is its *bias* in the additive scale.[3]

Definition 1 (Bias of ES with respect to Q)**.** *Let Q be a computable quantity given a* fully specified *linear structural causal model, and let ES be any estimand (a functional of the covariance matrix Σ). The* bias *of ES with respect to Q is the difference between the two quantities, $B = ES - Q$.*

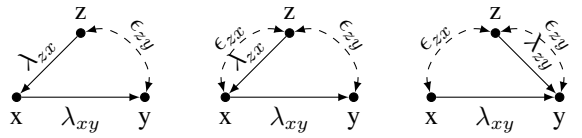
In our example, the proposed estimand is $ES = R_{yx.z}$, the target quantity is $Q = \lambda_{xy}$, and to compute the bias, $B = R_{yx.z} - \lambda_{xy}$, one needs to *identify* λ_{xy}. Computing the bias, thus, entails an identification problem (Prop. 1).

Proposition 1. *The bias of estimand ES with respect to target quantity Q is identifiable iff Q is identifiable.*

In G_A, however, the presence of the bidirected edge $x \leftrightarrow z$ renders λ_{xy} *unidentifiable*, and computation of B is not possible. How could one circumvent this impediment?

As in Cornfield et al. (1959), the impossibility of computing the exact bias of $R_{yx.z}$ with respect to λ_{xy} calls for another strategy—expressing the bias as a function of the "strength" of the omitted confounders. In this way, the analyst can predict for any *hypothetical* strength of the confounders whether it would be enough to change the research conclusions. This allows the analyst to bring new substantive knowledge to bear, by submitting these quantitative results to a judgment of plausibility and ruling out some scenarios.

[3]Note this refers to the bias of an *estimand* (*not* an estimator), and it is the difference between the proposed estimand and the desired (causal) target quantity in the *population*.



(a) Model G_O (b) Model G_A (c) Model G_B

Figure 1: Original model G_O and two alternative models, G_A and G_B. In G_A any of the remaining parameters (λ_{zx}, ϵ_{zx} or ϵ_{zy}) can be used as a sensitivity parameter for λ_{xy}, whereas G_B rules out ϵ_{zx} as a sensitivity parameter. Adding a bidirected edge $x \leftrightarrow y$ in G_A does not prevent ϵ_{zy} from being a valid sensitivity parameter, whereas in G_B it does.

Implementing this idea requires a precise definition of how to measure the "strength" of the omitted confounders. In our example, a possible candidate for measuring such strength is the structural parameter ϵ_{zx} of the added bidirected edge $z \leftrightarrow x$. The task then becomes: (i) to determine whether knowledge of ϵ_{zx} allows the identification of λ_{xy}; and, (ii) if so, to find a parameterized estimand for λ_{xy} in terms of ϵ_{zx}. This 2-step procedure can be seen as an identification problem with non-zero constraints (Def. 2).[4]

Definition 2 (θ-identifiability)**.** *Let M be a linear SCM and θ a set of parameters of M with known (non-zero) values. A causal quantity Q is said to be θ-identifiable if Q is uniquely computable from Σ and θ.*

We call any functional of Σ and θ, which *identifies* Q, a *θ-specific estimand* (or sensitivity curve) for Q with respect to *sensitivity parameters* θ. These estimands are the workhorse for sensitivity analysis; they allow us to investigate how strong certain relationships must be (as parameterized by θ) in order to induce significant bias in our estimates. In other words, identifying a bias function in terms of θ (and the observed data) for sensitivity analysis is equivalent to the constrained identification problem of Def. 2 (Prop. 2).

Proposition 2. *The bias of ES with respect to Q can be expressed as a function of θ (and Σ) iff Q is θ-identifiable.*

Going back to G_A, it is indeed possible to construct an ϵ_{zx}-specific estimand for λ_{xy} (see Sec. 4):

$$\lambda_{xy}(\epsilon_{zx}) = \frac{\sigma_{xy} - (\sigma_{zx} - \epsilon_{zx})\sigma_{yz}}{1 - (\sigma_{zx} - \epsilon_{zx})\sigma_{zx}} \qquad (1)$$

Eq. 1 allows one to compute the bias of $R_{yx.z}$ with respect to the target quantity λ_{xy}, for any given hypothetical value of ϵ_{zx}, if the true model were given by G_A. Similarly, it allows one to determine how strong the unobserved confounder would need to be (as parameterized by ϵ_{zx}) such that the as-

[4]Note the relationship to z-ID (Bareinboim & Pearl, 2012), in which case constraints are imposed on experimental distributions in the non-parametric setting.

sociation $R_{yx.z}$ is completely explained by the unobserved confounder (i.e., the value of ϵ_{zx} such that $\lambda_{xy}(\epsilon_{zx}) = 0$).

Still, what if the analyst has no knowledge to plausibly bound the strength of ϵ_{zx}? Even though the violation introduced in model G_A was the addition of the bidirected edge $x \leftrightarrow z$, corresponding to ϵ_{zx}, there is no reason to limit our attention to that parameter, and *any* θ-specific estimand could be used for sensitivity analysis. In fact, the two remaining parameters of the model also yield valid θ-specific estimands (Sec. 5 provides an algorithmic solution),

$$\lambda_{xy}(\lambda_{zx}) = \frac{\sigma_{xy} - \lambda_{zx}\sigma_{yz}}{1 - \lambda_{zx}\sigma_{zx}} \tag{2}$$

$$\lambda_{xy}(\epsilon_{zy}) = \frac{\sigma_{zy} - \epsilon_{zy}}{\sigma_{zx}} \tag{3}$$

Having a diverse option of sensitivity curves is important, because sensitivity analysis relies on plausibility judgments. One could argue, for instance, that assessing the plausibility of ϵ_{zx} could be hard because it involves judging the effect of confounders of *unknown* cardinality, and perhaps, previous studies give plausible bounds on the direct causal effect of z on x (i.e., λ_{zx}), making a λ_{zx}-specific estimand more attractive. Regardless of the specific scenario, it is clear that the choice of sensitivity parameters should be guided by the availability of substantive knowledge.

Remarkably, several subtleties arise when deriving θ-specific estimands, even in simple models with three variables. For instance, a natural approach for tackling the problem in our example could be the re-expression of $R_{yx.z}$ in terms of the covariance matrix implied by G_A, yielding,

$$R_{yx.z} = \lambda_{xy} - \frac{(\sigma_{zx} - \lambda_{zx})\epsilon_{zy}}{1 - \sigma_{zx}^2} = \lambda_{xy} - \frac{\epsilon_{zx}\epsilon_{zy}}{1 - \sigma_{zx}^2} \tag{4}$$

One may surmise upon the examination of such expression that two sensitivity parameters are needed. As shown in Eqs. 1 to 3, this conclusion would be misleading.

These subtleties also appear when solving several variations of a model. Imagine the alternative model is now G_B, instead of G_A, as shown in Figure 1c. Is ϵ_{zx} an admissible sensitivity parameter in this case? Is the ϵ_{zy}-specific estimand derived in G_A still valid if the model were G_B? If we include another violation in both models, a bidirected arrow $x \leftrightarrow y$, would the previously obtained ϵ_{zy}-specific estimands still be valid? Despite the apparent similarity of both models, the answers to these questions reveal their sensitivity curves behave quite differently. The tools developed in this paper not only provide an algorithmic solution to these questions, but also allow researchers to swiftly answer them by simple inspection of the graph.

The above examples demonstrate several of the identification problems entailed by a sensitivity analysis. If in small models these tasks are already complex, once we move to models with more than three or four variables, an informal, case-by-case approach to sensitivity analysis is simply infeasible. Therefore, we need a formal framework and efficient algorithms to incorporate constraints in linear SCMs.

4. Incorporating constraints in linear SCMs

Existing methods for identification in linear SCMs, such as the QID algorithm from Chen et al. (2017), are able to incorporate constraints on directed edges and can be used to derive sensitivity curves such as the λ_{zx}-specific estimand of Eq. 2. The QID algorithm exploits a known edge λ_{ab} by creating an auxiliary variable (AV) $b^* = b - \lambda_{ab}a$ (Chen et al., 2016). Subtracting out the direct effect of a on b in this way may help with the identification of other coefficients in the model. For instance, the λ_{zx}-specific estimand can be computed using AVs in the following way: (i) create $x^* = x - \lambda_{zx}z$; (ii) use x^* as an instrument for λ_{xy}, resulting in $\lambda_{xy}(\lambda_{zx}) = \sigma_{yx^*}/\sigma_{xx^*} = (\sigma_{xy} - \lambda_{zx}\sigma_{yz})/(1 - \lambda_{zx}\sigma_{zx})$.[5]

However, neither the ϵ_{zx}-specific nor the ϵ_{zy}-specific estimands can be derived using QID; in fact, there is no current identification algorithm that offers a principled and efficient way to exploit knowledge of bidirected edges.[6] As this is critical for the derivation of sensitivity curves (see Sec. 6), one of the core contributions of this work is the development of a novel graphical procedure that allows one to systematically incorporate constraints on error covariances.

Conventional linear SCMs already impose one type of constraint on error covariances: a lack of a bidirected edge between two variables a and b encodes the assumption that the structural parameter ϵ_{ab} is zero. The identification problem imposed by sensitivity analysis, nonetheless, sets a different type of constraint—the error covariance ϵ_{ab} is fixed to a *known* but *non-zero* number. The essence of our method is to represent this knowledge in the graph.

Considering a graph G, covariance matrix Σ, and a *known* error covariance ϵ_{ab}, our strategy consists of performing a "manageable" transformation of G such that the bidirected edge $a \leftrightarrow b$ is removed from the graph. By "manageable" we mean the implied covariance matrix Σ' of the transformed graph G' can still be derived from Σ and the known value ϵ_{ab}; otherwise, we would have no connection between G' and the data, making inference in G' impossible. Once this graphical transformation is applied, we can exploit *any*

[5]The QID algorithm extends generalized instrumental sets (Brito & Pearl, 2002) using a bootstrapping procedure whereby complex models can be identified by iteratively identifying coefficients and using them to generate new auxiliary variables. It takes as inputs a graph G, covariance matrix Σ and *known* directed edges $\mathcal{D}$, and it returns the new set of identified directed edges.

[6]Methods from computer algebra offer a complete solution but are computationally intractable. See Sec. 6 and Supp. Materials.

existing graphical identification method on the modified model G', and solutions in G' can be transfered back to solutions in the original model G. In short, we manipulate the graph to reduce an identification problem with a non-zero constraint to a standard one.

The easiest way to introduce our method, which we call PUSHFORWARD, is via an example. Consider again graph G_A in Fig. 1b, and assume ϵ_{zy} is known. Path-tracing (Wright, 1921) results in the following covariances, where the known parameter ϵ_{zy} is highlighted in red,

$$\sigma_{zx} = \lambda_{zx} + \epsilon_{zx} \tag{5}$$
$$\sigma_{zy} = \lambda_{zx}\lambda_{xy} + \epsilon_{zx}\lambda_{xy} + \epsilon_{zy} \tag{6}$$
$$\sigma_{xy} = \lambda_{xy} + \lambda_{zx}\epsilon_{zy} \tag{7}$$

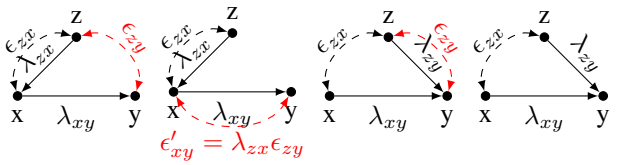
Ideally, we could create an alternative model G^*_A where the bidirected edge $z \leftrightarrow y$ is fully removed from the graph. For this to be useful, we need to be able to express the new implied covariance matrix Σ^*_A in terms of the original covariance matrix Σ_A and the known error covariance ϵ_{zy}. While expressing σ^*_{zy} in terms of Σ_A and ϵ_{zy} is straightforward (since, trivially, $\sigma^*_{zy} = \sigma_{zy} - \epsilon_{zy}$), it is not immediately clear how to write $\sigma^*_{xy} = \sigma_{xy} - \lambda_{zx}\epsilon_{zy} = \lambda_{xy}$ in terms of Σ_A and ϵ_{zy}, for this requires identifying either λ_{xy} or λ_{zx} in the original model.

Thus, rather than fully removing $z \leftrightarrow y$, we "push it forward" to the children of z, as shown in graph G'_A of Fig. 2b. Note the bidirected edge is moved from being between z and y to being between x (a child of z) and y, with new structural parameter $\epsilon'_{zy} = \lambda_{zx}\epsilon_{zy}$. Path-tracing of G'_A shows its implied covariance matrix Σ'_A is exactly the same as Σ_A, except for σ'_{zy}, which can be obtained by subtracting ϵ_{zy} from σ_{zy},

$$\sigma'_{zx} := \sigma_{zx} \qquad = \lambda_{zx} + \epsilon_{zx} \tag{8}$$
$$\sigma'_{zy} := \sigma_{zy} - \epsilon_{zy} \qquad = \lambda_{zx}\lambda_{xy} + \lambda_{xy}\epsilon_{zx} \tag{9}$$
$$\sigma'_{xy} := \sigma_{xy} \qquad = \lambda_{xy} + \lambda_{zx}\epsilon_{zy} \tag{10}$$

Since G'_A has the same structural coefficients as G and we know how to compute the covariance matrix induced by G'_A from the known values Σ and ϵ_{zy}, we can use G'_A to identify the coefficients in our original model. In this case, z is an *instrument* for λ_{xy} in G'_A, resulting in the estimand $\lambda_{xy}(\epsilon_{zy}) = \sigma'_{zy}/\sigma'_{zx} = (\sigma_{zy} - \epsilon_{zy})/\sigma_{zx}$ of Eq. 3.

Applying the same logic to graph G_B in Fig. 1c, assume ϵ_{zy} is known. Since z has no other descendants except y, pushing forward ϵ_{zy} simply removes the bidirected edge $z \leftrightarrow y$. This results in the modified graph G'_B of Fig. 2d with the amortized covariance of z and y, $\sigma'_{zy} = \sigma_{zy} - \epsilon_{zy}$. Note ϵ_{zy} enters in no other covariances of the system. The graph G'_B renders z single-door admissible for the identification of λ_{xy}, giving us the estimand $\lambda_{xy}(\epsilon_{zy}) = R'_{yx.z} = (\sigma_{yx} - \sigma_{xz}(\sigma_{zy} - \epsilon_{zy}))/(1 - \sigma^2_{xz})$.

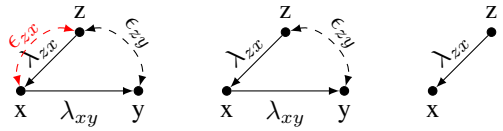


(a) G_A (b) G'_A (c) G_B (d) G'_B

Figure 2: Pushing forward ϵ_{zy} in G_A renders z a valid instrument in G'_A. Pushing forward ϵ_{zy} in G_B renders z single-door admissible in G'_B.

This simple graphical manipulation also makes it clear why adding a bidirected edge $x \leftrightarrow y$ as a further violation in the original graphs G_A and G_B has different consequences for the identification of λ_{xy}. In G'_A, z still remains a valid instrument even if the original graph had $x \leftrightarrow y$; this would only change the value of the structural coefficient ϵ'_{xy}, which would now read $\epsilon'_{xy} = \epsilon_{xy} + \lambda_{zx}\epsilon_{zy}$. In G'_B, however, adding $x \leftrightarrow y$ renders z inadmissible for single-door identification of λ_{xy}, since this backdoor path cannot be blocked.

Sometimes it might be necessary to prune variables from G' to guarantee Σ' is computable. Consider again G_A and assume ϵ_{zx} is known. Pushing forward ϵ_{zx} results in Fig. 3b where, as before, we know $\sigma'_{zx} = \sigma_{zx} - \epsilon_{zx}$. However, path-tracing of Fig. 3b shows the covariance of z with y would also need adjustment, $\sigma'_{zy} = \sigma_{zy} - \lambda_{xy}\epsilon_{zx}$. Thus, we have two cases: (i) if λ_{xy} is known, the adjustment is feasible and we are done; (ii) if λ_{xy} is not (yet) known, the adjustment cannot be made; but, since y is a leaf node, it can be pruned from G' (Tian & Pearl, 2003), avoiding this problem (Fig. 3c). In this case, note the pruned graph is still helpful—now λ_{zx} can be identified. As previously discussed, knowledge of λ_{zx} permits identification of λ_{xy} using AVs, giving us the ϵ_{zx}-specific estimand of Eq. 1.



(a) G_A (b) Push forward ϵ_{zx} (c) Prune y

Figure 3: Pushing forward ϵ_{zx} in G_A requires adjusting σ_{yz}. If the adjustment is possible, y is kept in the graph as in Fig. 3b; if not, y is marginalized (pruned) as in Fig. 3c.

The graphical manipulation of PUSHFORWARD is general, and can be performed whenever we have knowledge of a known error covariance. Theorem 1 formalizes the procedure to arbitrary models. Given any bidirected edge $x \leftrightarrow y$ with known value ϵ_{xy}, we remove it from the graph and register the new amortized covariance $\sigma'_{xy} = \sigma_{xy} - \epsilon_{xy}$. Next we repair the covariances of the descendants of x with y by,

for every $c \in Ch(x)$, adding (or modifying) the bidirected edge $c \leftrightarrow y$ with the direct causal effect λ_{xc} times ϵ_{xy}. Finally, for any descendant z of y, we either (i) amortize its covariance with x, if *all* edges that compose the total causal effect δ_{yz} of y on its descendant z are known, or (ii) marginalize z out by pruning the graph. The final output is a modified model $\langle G', \Sigma' \rangle$ where any graphical identification method can be applied; and, estimands in terms of Σ' can be converted back to estimands in terms of Σ and ϵ_{xy}.

Theorem 1 (PUSHFORWARD)**.** *Given a linear SCM with graph G, covariance matrix Σ, a set of known directed edges $\mathcal{D}$, and known bidirected edge ϵ_{xy}, let the pair $\langle G', \Sigma' \rangle$ be constructed from G and Σ as follows:*

1. *$x \leftrightarrow y$ is removed and $\sigma'_{xy} = \sigma_{xy} - \epsilon_{xy}$;*
2. *$\forall c \in Ch(x), c \neq y$, the bidirected edges $c \leftrightarrow y$ are added if they do not exist, and $\epsilon'_{cy} = \epsilon_{cy} + \lambda_{cy}\epsilon_{xy}$;*
3. *$\forall z \in De(y), z \neq x$, if there is an edge on any directed path from y to z that is not in $\mathcal{D}$, then z is removed from G'. For the remaining z, $\sigma'_{xz} = \sigma_{xz} - \epsilon_{xy}\delta_{yz}$, where δ_{yz} is the sum of all directed paths from y to z;*
4. *All other parameters and covariances remain the same.*

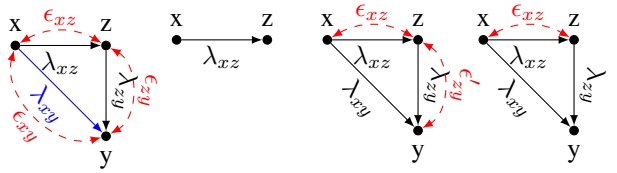
Then, if λ_{ab} is identifiable in G', it is $(\epsilon_{xy}, \mathcal{D})$-identifiable in G.

We denote by $\text{PF}(G, \Sigma, \mathcal{D}, \epsilon_{xy}, x)$ the function that returns the modified model $\langle G', \Sigma' \rangle$ as per Theorem 1. Pseudocode for PF (which closely follows the steps of the theorem) as well as the proof can be found in the supplementary material.

5. Algorithmic derivation of sensitivity curves

In this section, we construct a graph-based constrained identification algorithm for linear SCMs which systematically exploits knowledge of *both* path coefficients and error covariances efficiently. Our algorithm relies on the PUSHFORWARD method to incorporate constraints on bidirected edges, and on the AV technique (via the QID algorithm) to incorporate constraints on directed edges. This allows the algorithmic derivation of sensitivity curves for a target query λ_{xy} in arbitrary linear models, with an arbitrary set of directed and bidirected edges as sensitivity parameters.

Although the graphical modification of PUSHFORWARD is defined for one bidirected edge, the modified graph G' is a valid model in which any graphical operation can be performed. We can thus extend PUSHFORWARD to handle multiple bidirected edges by iteratively applying it whenever a bidirected edge of the modified graph is still known—what remains to be decided is the order in which to perform these operations. Note that testing all possible orders of graphical

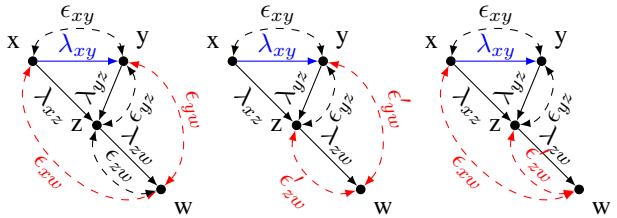


(a) G (b) G'_z (c) G'_y (d) G''_y

Figure 4: PF multiple edges in topological order.

manipulations can result in an algorithm with exponential computational complexity, even when initially pushing forward a single bidirected edge ϵ_{xy}. This happens because new bidirected edges are created for each $c \in Ch(x)$ and, if all the λ_{cx} are identifiable, all subsets of those bidirected edges may be eligible to be pushed forward again. Thus, here we propose an efficient procedure using topological ordering, which performed *as well* as a brute-force approach in our computational experiments (Sec. 6).

Consider the example given in Fig. 4a. The task is to decide whether $\theta = (\epsilon_{xz}, \epsilon_{xy}, \epsilon_{zy})$ (in red) is an admissible set of sensitivity parameters for the target coefficient λ_{xy} (in blue) and, if so, to find the corresponding sensitivity curve. Our strategy consists of, for each node v, listing its ancestors $a \in An(v)$, and, in topological order, iteratively push forward ϵ_{av} if it is still known in the modified graph. By performing operations in this way, we are guaranteed to visit each ancestor of v only once. Starting with node $v = z$, it has only one ancestor x and a single known bidirected edge to be removed, ϵ_{xz}. This can be handled with a one-step PUSHFORWARD operation (pruning y), resulting in the modified graph G'_z of Fig. 4b, in which λ_{xz} can be trivially identified. Next, return to the original graph and consider $v = y$, with ancestors x and z. Following a topological order, we first push forward ϵ_{xy}, giving us the modified graph G'_y of Fig. 4c with new bidirected edge $\epsilon'_{zy} = \epsilon_{zy} + \lambda_{xz}\epsilon_{xy}$. Note all components of ϵ'_{zy} are known, we can thus push forward ϵ'_{zy} in G'_y, obtaining the graph G''_y in Fig. 4d, in which λ_{xy} is identified with sensitivity curve $R''_{yx.z}$.



(a) Original Model (b) Push forward ϵ_{xw} (c) Push forward ϵ_{yw}

Figure 5: Instruments with ancestors and descendants.

In the previous example we demonstrated how to systematically deal with bidirected edges connected to *ancestors*

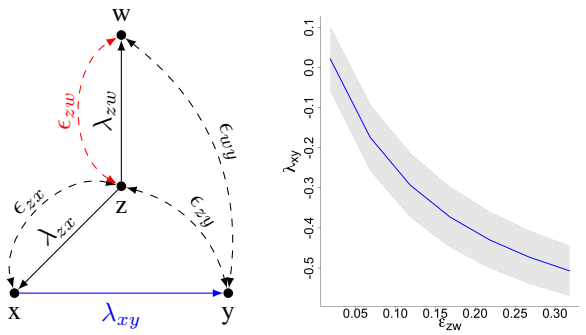
Algorithm 1 cID$(G, \Sigma, \mathcal{D}, \mathcal{B})$

1: **initialize** $V_{\mathcal{B}} \leftarrow$ Vertices$(\mathcal{B})$
2: **repeat**
3: $\quad\mathcal{D} \leftarrow \mathcal{D} \cup$ QID$(G, \Sigma, \mathcal{D})$
4: $\quad$**for each** $v \in V_{\mathcal{B}}$ **do**
5: $\quad\quad\langle G', \Sigma' \rangle \leftarrow \langle G, \Sigma \rangle$
6: $\quad\quad$**for each** $a \in An(v)$ in topological order **do**
7: $\quad\quad\quad$**if** ϵ'_{av} is known **then**
8: $\quad\quad\quad\quad\langle G', \Sigma' \rangle \leftarrow$ PF$(G', \Sigma', \mathcal{D}, \epsilon'_{av}, a)$
9: $\quad\quad\quad\quad\mathcal{D} \leftarrow \mathcal{D} \cup$ QID$(G', \Sigma', \mathcal{D})$
10: $\quad\quad\quad$**end if**
11: $\quad\quad$**end for**
12: $\quad$**end for**
13: $\quad$**for each** $\epsilon_{ab} \in \mathcal{B}$ **do**
14: $\quad\quad\langle G', \Sigma' \rangle \leftarrow$ PF$(G, \Sigma, \mathcal{D}, \epsilon_{ab}, a)$
15: $\quad\quad\mathcal{D} \leftarrow \mathcal{D} \cup$ QID$(G', \Sigma', \mathcal{D})$
16: $\quad$**end for**
17: **until** all directed edges have been identified or no edge has been identified in the last iteration

of a node v; however, in linear models, *descendants* of v can also help with the identification of direct causal effects λ_{av}. Consider, for instance, Fig. 5a. The task is to find a sensitivity curve for λ_{xy} in terms of $\theta = (\epsilon_{xw}, \epsilon_{yw})$. Start with node w and, as before, push forward ϵ_{xw} as in Fig. 5b. Here, λ_{zw} can be identified with x as an instrument. Returning to the original graph, now consider node y and push forward the bidirected edge ϵ_{yw} with its *descendant* w, as in Fig. 5c. Since λ_{zw} has been identified, we can create the AV $w^* = w - \lambda_{zw} z$ which is a valid instrument for λ_{xy}.

These two cases illustrate our general procedure for handling multiple bidirected edges, which in combination with the QID algorithm forms our constrained identification algorithm CID, provided in Algorithm 1. Lines 4 to 12 perform PUSHFORWARD (PF) in topological ordering, each time applying QID in the modified model to verify if new directed edges can be identified; lines 13 to 16 perform a single PUSHFORWARD operation on each bidirected edge, which may free descendants to be used as instruments as in Fig. 5. Since new identified edges can help both PUSHFORWARD as well as QID, this process is repeated until all or no new directed edges are identified in the last iteration. The complexity of CID is dominated by QID, which is polynomial if the degree of each node is bounded (Chen et al., 2017).

An interesting 4-node example is shown in Fig. 6a, where ϵ_{zw}, a parameter neither related to x nor y, is an admissible sensitivity parameter for λ_{xy}! Our algorithm derives an ϵ_{zw}-specific estimand for λ_{xy} as follows. It first pushes forward ϵ_{zw}, and runs QID in the modified graph, resulting in the identification of λ_{zw}. Next, the algorithm returns to the original graph, and runs QID, which uses λ_{zw} to create the auxiliary variable $w^* = w - \lambda_{zw} z$, enabling



(a) λ_{xy} is ϵ_{zw}-identifiable (b) Sensitivity of λ_{xy} in terms of ϵ_{zw}

Figure 6: In Fig 6a note that, although not connected to x nor y, ϵ_{zw} is an admissible sensitivity parameter for λ_{xy}. Fig. 6b shows the sensitivity curve of λ_{xy} in terms of ϵ_{zw} for a numerical simulation of the model in Fig. 6a.

the identification of λ_{zx}. Finally, still within QID, λ_{xy} is obtained using the auxiliary variable $x^* = x - \lambda_{zx} z$.

As discussed in Sec.3, the utility of θ-specific estimands is to show how sensitive the target quantity of interest is to different hypothetical values of the sensitivity parameters θ. These results can then be submitted to quantitative plausibility judgments, for instance, in the form of $\theta \in \Theta_p$, where Θ_p is a plausibility region. To illustrate how one could deploy this in practice, we provide a numerical example of the causal model in Fig. 6a. Our goal is to assess how different hypothetical values for ϵ_{zw} affects inference of λ_{xy}. In a real context, this needs to be estimated from finite samples, and here we use a maximum likelihood estimator. Fig. 6b shows the estimates for λ_{xy} (blue) for different values of the sensitivity parameter ϵ_{zw}, along with the corresponding 95% confidence interval (gray). If, for instance, we can plausibly bound ϵ_{zw} to be within 0.1 to 0.3, the plot reveals λ_{xy} can be safely judged to be within -0.2 to -0.6.

6. Computational experiments

The identification problem in linear systems has not yet been efficiently solved. Although there exists a complete solution using computer algebra (García-Puente et al., 2010), these methods are computationally intractable, making it impractical for graphs larger than 4 or 5 nodes. Since we rely on existing identification algorithms that are polynomial but not complete (i.e., QID cannot find all identifiable parameters), we cannot expect the CID algorithm to find all sensitivity curves as well. In this section, we report the results of an extensive set of experiments aimed to empirically verify the generality of our approach. We have performed an exhaustive study of all possible queries in 3 and 4-node models, which are essentially the largest instances computer

ID Algorithm	3-Node Models			4-Node Models		
	Directed	*Bidirected*	*Both*	*Directed*	*Bidirected*	*Both*
QID (AVs only)	19(100%)	– (0%)	68 (21%)	14,952(95%)	– (0%)	170,304(29%)
CID (AVs + PF)	19(100%)	109(100%)	320(100%)	14,952(95%)	50,708(97%)	555,758(96%)
Ground Truth	19	109	320	15,740	52,016	578,858

Table 1: Number of θ-identifiable sensitivity queries (only when $\theta \neq \emptyset$) per type of sensitivity parameters θ.

algebra methods can solve through brute force.[7]

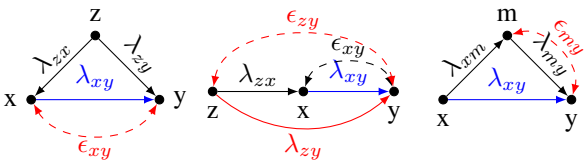
A query consists of determining whether in model G, a target parameter λ_{xy} is θ-identifiable given a set of sensitivity parameters θ. For 3-node models, we have 50 connected graphs with 720 possible queries; for 4-node models, we have 3,745 connected graphs and 1,059,156 possible queries.[8] For each query, we used algebraic methods to determine ground-truth identification and checked it against the results of both QID and CID. Our interest lies in the queries that are θ-identifiable *only* when $\theta \neq \emptyset$.

The results are given in Table 1, where columns restrict sensitivity parameters θ to be: (i) subsets of directed edges; (ii) subsets of bidirected edges; and, (iii) subsets of both directed and bidirected edges. The results show that our CID algorithm correctly identifies all possible sensitivity curves for 3-node models. Among 4-node models, our method solves 96% of all identifiable sensitivity queries.

These numbers reveal that, in the context of linear SCMs, canonical sensitivity analysis examples which have been addressed on a case-by-case basis in the literature (e.g., Fig.7, target coefficient in blue and sensitivity parameters in red), are only a *small* subset of all possible sensitivity analyses exercises enabled by our proposal.When comparing CID's results to those of QID only, it is also clear that systematically incorporating constraints on bidirected edges is essential for obtaining sensitivity curves.

A valid concern regarding CID's current implementation is that the proposed topological ordering for processing bidirected edges could be less capable than a general search over all possible valid graphical manipulations. With this in mind, we performed a thorough comparison of our proposal against other ordering methods for all queries in 3 and 4-node models. Topological ordering proved to perform *as well* as a brute-force search that recursively tests all possible subsets of bidirected edges that can be pushed forward.

Finally, the incompleteness of CID can stem from two sources: limitations of the graphical manipulations performed by PushForward and the incompleteness of the identification algorithm for directed edges, QID. Separating the two can help guide efforts for future research. To achieve that, we used algebraic methods to simulate how CID would have performed if it had access to a complete identification algorithm for directed edges instead of QID. We found that CID would have identified over 99.99% of 4-node sensitivity queries. This seem to suggest that: (i) the main bottleneck of CID is QID; and (ii) PushForward with topological ordering can reap the benefits of improved identification algorithms for directed edges.



(a) Backdoor (b) IV (c) Mediation

Figure 7: Canonical sensitivity analyses: (a) backdoor violation with unobserved confounders independent of observed confounders (Carnegie et al., 2016); (b) putative instrumental variable, where both the exclusion and independence restriction are suspected to be violated (Wang et al., 2018); (c) randomized trial in which treatment x has side-effect m, and unobserved mediation-outcome confounding cannot be ruled out (VanderWeele, 2010). For linear SCMs, these are special cases of all queries solved by our approach.

7. Conclusion

We introduced a general algorithmic framework of sensitivity analysis for linear SCMs. We reduced sensitivity analysis to a constrained identification problem and developed a novel graphical procedure to systematically incorporate constraints on bidirected edges. We then devised an efficient graph-based algorithm for deriving sensitivity curves. Exhaustive experiments corroborated the generality of our proposal. Such systematic tools can help analysts better navigate in the model space and understand the trade-off between the plausibility of assumptions and the strength of conclusions. Extensions to other types of violations and to nonlinear models are promising directions for future work.

[7] We use Gröbner bases, which has a doubly-exponential comp. complexity (Bardet, 2002). See Supp. Materials for details.

[8] For 5-node models, these numbers reach 1 million graphs and 11 *billion* queries. Ground-truth computations in 5-node models using computer algebra can take hours for a *single* graph.

Acknowledgements

The authors thank anonymous reviewers for helpful comments. Cinelli and Pearl are supported in parts by grants from IBM Research, Defense Advanced Research Projects Agency [W911NF-16-057], National Science Foundation [IIS-1302448, IIS-1527490, and IIS-1704932], and Office of Naval Research [N00014-17-S-B001]. Bareinboim and Kumor are supported in parts by grants from NSF IIS1704352, and IIS-1750807 (CAREER), IBM Research, and Adobe Research. Most of the work by Chen was conducted while at IBM Research AI.

References

Arah, O. A. Bias analysis for uncontrolled confounding in the health sciences. *Annual review of public health*, 38: 23–38, 2017.

Bardet, M. On the complexity of a grobner basis algorithm. pp. 8, 2002.

Bareinboim, E. and Pearl, J. Causal inference by surrogate experiments: z-identifiability. *arXiv preprint arXiv:1210.4842*, 2012.

Bareinboim, E. and Pearl, J. Causal inference and the data-fusion problem. *Proceedings of the National Academy of Sciences*, 113(27):7345–7352, 2016.

Blackwell, M. A selection bias approach to sensitivity analysis for causal effects. *Political Analysis*, 22(2):169–182, 2013.

Brito, C. and Pearl, J. Generalized instrumental variables. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pp. 85–93. Morgan Kaufmann Publishers Inc., 2002.

Brumback, B. A., Hernán, M. A., Haneuse, S. J., and Robins, J. M. Sensitivity analyses for unmeasured confounding assuming a marginal structural model for repeated measures. *Statistics in medicine*, 23(5):749–767, 2004.

Carnegie, N. B., Harada, M., and Hill, J. L. Assessing sensitivity to unmeasured confounding using a simulated potential confounder. *Journal of Research on Educational Effectiveness*, 9(3):395–420, 2016.

Chen, B., Pearl, J., and Bareinboim, E. Incorporating knowledge into structural equation models using auxiliary variables. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*, pp. 3577–3583, 2016.

Chen, B., Kumor, D., and Bareinboim, E. Identification and model testing in linear structural equation models using auxiliary variables. In *International Conference on Machine Learning*, pp. 757–766, 2017.

Cinelli, C. and Hazlett, C. Making sense of sensitivity: Extending omitted variable bias, 2018.

Cornfield, J., Haenszel, W., Hammond, E. C., Lilienfeld, A. M., Shimkin, M. B., and Wynder, E. L. Smoking and lung cancer: recent evidence and a discussion of some questions. *journal of National Cancer Institute*, (23): 173–203, 1959.

Cox, D., Little, J., and O'shea, D. *Ideals, Varieties, and Algorithms*, volume 3. Springer, 1992.

Ding, P. and VanderWeele, T. J. Sensitivity analysis without assumptions. *Epidemiology (Cambridge, Mass.)*, 27(3): 368, 2016.

Fisher, R. Cigarettes, cancer, and statistics. *The Centennial Review of Arts & Science*, 2:151–166, 1958.

Fisher, R. A. Dangers of cigarette-smoking. *British Medical Journal*, 2(5039):297, 1957.

Foygel, R., Draisma, J., and Drton, M. Half-trek criterion for generic identifiability of linear structural equation models. *The Annals of Statistics*, pp. 1682–1713, 2012.

Frank, K. A. Impact of a confounding variable on a regression coefficient. *Sociological Methods & Research*, 29 (2):147–194, 2000.

Franks, A., D'Amour, A., and Feller, A. Flexible sensitivity analysis for observational studies without observable implications. *Journal of the American Statistical Association*, (just-accepted):1–38, 2019.

García-Puente, L. D., Spielvogel, S., and Sullivant, S. Identifying causal effects with computer algebra. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence (UAI). AUAI Press*, 2010.

Giffin, R. B., Lebovitz, Y., English, R. A., et al. *Transforming clinical research in the United States: challenges and opportunities: workshop summary*. National Academies Press, 2010.

Imai, K., Keele, L., and Yamamoto, T. Identification, inference and sensitivity analysis for causal mediation effects. *Statistical science*, pp. 51–71, 2010.

Imbens, G. W. Sensitivity to exogeneity assumptions in program evaluation. *The American Economic Review*, 93 (2):126–132, 2003.

Joffe, M. M., Yang, W. P., and Feldman, H. I. Selective ignorability assumptions in causal inference. *The International Journal of Biostatistics*, 6(2), 2010.

Leamer, E. Let's take the con out of econometrics. *The American Economic Review*, 73(1):31–43, 1983.

Masten, M. A. and Poirier, A. Identification of treatment effects under conditional partial independence. *Econometrica*, 86(1):317–351, 2018.

Mauro, R. Understanding love (left out variables error): A method for estimating the effects of omitted variables. *Psychological Bulletin*, 108(2):314, 1990.

Oster, E. Unobservable selection and coefficient stability: Theory and evidence. *Journal of Business & Economic Statistics*, pp. 1–18, 2017.

Pearl, J. *Causality*. Cambridge university press, 2000.

Rosenbaum, P. R. *Design of observational studies*. Springer Series in Statistics, 2010.

Rosenbaum, P. R. and Rubin, D. B. Assessing sensitivity to an unobserved binary covariate in an observational study with binary outcome. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 212–218, 1983.

Small, D. S. Sensitivity analysis for instrumental variables regression with overidentifying restrictions. *Journal of the American Statistical Association*, 102(479):1049–1058, 2007.

Spirtes, P., Glymour, C. N., Scheines, R., Heckerman, D., Meek, C., Cooper, G., and Richardson, T. *Causation, prediction, and search*. MIT press, 2000.

The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 8.5)*, 2018. `https://www.sagemath.org`.

Tian, J. and Pearl, J. On the identification of causal effects. Technical Report R-290, Cognitive Systems Laboratory, UCLA, 2003.

VanderWeele, T. J. Bias formulas for sensitivity analysis for direct and indirect effects. *Epidemiology (Cambridge, Mass.)*, 21(4):540, 2010.

Vanderweele, T. J. and Arah, O. A. Bias formulas for sensitivity analysis of unmeasured confounding for general outcomes, treatments, and confounders. *Epidemiology (Cambridge, Mass.)*, 22(1):42–52, January 2011. ISSN 1531-5487.

Wang, X., Jiang, Y., Zhang, N. R., and Small, D. S. Sensitivity analysis and power for instrumental variable studies. *Biometrics*, 2018.

Wright, S. Correlation and causation. *Journal of agricultural research*, 20(7):557–585, 1921.

Appendix

A. Proof of propositions 1 and 2

The propositions follow directly from the definitions, but we state the proofs here for completeness. For proposition 1, first note ES is a functional of the covariance matrix Σ and it is by definition identifiable. Thus, if Q is identifiable, we can also uniquely compute Q from Σ and, since $B = ES - Q$, and each of its components are identifiable, B can also be uniquely computed from Σ and it is thus identifiable. Conversely, if B is identifiable, just write $Q = ES + B$, which means Q can be uniquely determined from Σ and it is also identifiable.

Proposition 2 follows the same argument. First note that if Q is θ-identifiable then we can write $B(\theta) = ES - Q(\theta)$ which is uniquely determined by Σ and θ, giving us a bias function parameterized in terms of θ. Conversely, if there exists a function $B(\theta)$ which, by definition, gives us a unique bias in terms of θ (and the data Σ), we can write $Q(\theta) = ES + B(\theta)$. This implies Q can be uniquely determined from Σ and θ and it is thus θ-identifiable.

B. Proof and pseudocode for Theorem 1

Theorem 1 (PUSHFORWARD)**.** *Given a linear SCM with graph G, covariance matrix Σ, a set of known directed edges $\mathcal{D}$, and known bidirected edge ϵ_{xy}, let the pair $\langle G', \Sigma' \rangle$ be constructed from G and Σ as follows:*

1. *$x \leftrightarrow y$ is removed and $\sigma'_{xy} = \sigma_{xy} - \epsilon_{xy}$;*
2. *$\forall c \in Ch(x), c \neq y$, the bidirected edges $c \leftrightarrow y$ are added if they do not exist, and $\epsilon'_{cy} = \epsilon_{cy} + \lambda_{cy}\epsilon_{xy}$;*
3. *$\forall z \in De(y)$, $z \neq x$, if there is an edge on any directed path from y to z that is not in $\mathcal{D}$, then z is removed from G'. For the remaining z, $\sigma'_{xz} = \sigma_{xz} - \epsilon_{xy}\delta_{yz}$, where δ_{yz} is the sum of all directed paths from y to z;*
4. *All other parameters and covariances remain the same.*

Then if λ_{ab} is identifiable in G' it is $(\epsilon_{xy}, \mathcal{D})$-identifiable in G.

Before moving forward, we use a couple definitions from the literature, which make reasoning about paths in the graph easier:

Definition 3. *(Foygel et al., 2012) A path π from v to w is a **trek** if it has no colliding arrowheads, that is, π is of the*

form:

$$v \leftarrow ... \leftarrow \leftrightarrow \rightarrow ... \rightarrow w$$
$$v \leftarrow ... \leftarrow k \rightarrow ... \rightarrow w$$
$$v \leftarrow ... \leftarrow w$$
$$v \rightarrow ... \rightarrow w$$

Definition 4. *(Foygel et al., 2012) A **trek monomial** $\pi(\Lambda, \mathcal{E})$ for trek π is defined as the product of the structural parameters along the trek, multiplied by the trek's top error term covariance.*

In particular, if π does not contain a bidirected edge[9]*,*

$$\pi(\Lambda, \mathcal{E}) = \epsilon_k^2 \prod_{x \rightarrow y \in \pi} \lambda_{xy}$$

where k is the node at the "top" of the trek (it has no incoming edges). If the trek contains bidirected edge ϵ_{ab}, then

$$\pi(\Lambda, \mathcal{E}) = \epsilon_{ab} \prod_{x \rightarrow y \in \pi} \lambda_{xy}$$

Lemma 1. *(Foygel et al., 2012) The covariance between v and w, σ_{vw} can be written as the sum of the trek monomials of all treks between v and w ($\mathcal{T}_{vw}$):*

$$\sigma_{vw} = \sum_{\pi \in \mathcal{T}_{vw}} \pi(\Lambda, \mathcal{E})$$

At its core, identifiability of an edge λ in linear Gaussian SCM can be reduced to the problem of finding whether there exists a unique solution for λ in terms of covariances in the system of equations defined by the rules of path analysis (Foygel et al., 2012), and knowledge of existing directed and bidirected effects.

With this in mind, we can prove PUSHFORWARD.

Proof. Specified in the theorem is a covariance matrix Σ, a graph of the structural equations G, a set of known directed edges $\mathcal{D}$, and known bidirected edge ϵ_{xy}.

The system of equations constraining values of structural parameters is

$$\sigma_{vw} = \sum_{\pi \in \mathcal{T}_{vw}} \pi(\Lambda, \mathcal{E}) \qquad \forall v, w \in G$$

We first look at σ_{xy}, and define a new known quantity σ'_{xy}:

$$\sigma_{xy} = \epsilon_{xy} + \sum_{\pi \in \mathcal{T}_{xy} \setminus \{\epsilon_{xy}\}} \pi(\Lambda, \mathcal{E})$$
$$\sigma'_{xy} = \sigma_{xy} - \epsilon_{xy} = \sum_{\pi \in \mathcal{T}_{xy} \setminus \{\epsilon_{xy}\}} \pi(\Lambda, \mathcal{E})$$

[9]Note also that we can have a trek from v to v, including a trek that takes no edges at all, which would be simply ϵ_v^2

Algorithm 2 PF - PUSHFORWARD

function PF($G, \Sigma, \mathcal{D}, \epsilon_{xy}, x$)
  initialize $\langle G', \Sigma' \rangle \leftarrow \langle G, \Sigma \rangle$
  update $\epsilon'_{xy} \leftarrow 0$ in G' and $\sigma'_{xy} \leftarrow \sigma'_{xy} - \epsilon_{xy}$ in Σ'
  for each $c \in Ch(x)$ **do**
    update $\epsilon'_{cy} \leftarrow \epsilon'_{cy} + \lambda_{xc}\epsilon_{xy}$ in G'
  end for
  for each $z \in De(y)$ **do**
    if $Edges(\delta_{yz}) \subseteq \mathcal{D}$ **then**
      update $\sigma'_{xz} = \sigma_{xz} - \epsilon_{xy}\delta_{yz}$
    else
      remove z from G'
    end if
  end for
  return $\langle G', \Sigma' \rangle$
end function

We also look at all descendants of y, ($z \in Z$) where the directed paths from y to z (δ_{yz}) are made entirely of known edges ($Edges(\delta_{yz}) \subseteq \mathcal{D}$). We define

$$\delta_{ab} = \frac{1}{\epsilon_a^2} \sum_{\pi \in \mathcal{T}^{\rightarrow}_{xy}} \pi(\Lambda, \mathcal{E})$$

where $\mathcal{T}^{\rightarrow}_{xy}$ represents the set of treks taking only directed edges from a to b: $a \rightarrow ... \rightarrow b$.

For each such descendant of y, z, we define the quantity σ'_{xz}

$$\sigma_{xz} = \delta_{yz}\epsilon_{xy} + \sum_{\pi \in \mathcal{T}_{xy} \setminus \mathcal{T}^{\rightarrow}_{\epsilon_{xy} yz}} \pi(\Lambda, \mathcal{E})$$
$$\sigma'_{xz} = \sigma_{xz} - \delta_{yz}\epsilon_{xy} = \sum_{\pi \in \mathcal{T}_{xy} \setminus \mathcal{T}^{\rightarrow}_{\epsilon_{xy} yz}} \pi(\Lambda, \mathcal{E})$$

Here, we used $\mathcal{T}^{\rightarrow}_{\epsilon_{xy} yz}$ to represent the treks starting from ϵ_{xy}, and continuing from y to x (half-treks from x to z using ϵ_{xy}).

Finally, we define $\sigma'_{vw} = \sigma_{vw}$ for all other covariances between nodes a and b where both a and b are either non-descendants of y, or have their paths to y known.

This gives us a new system of equations in the original variables. All that remains to be shown is that an identified quantity λ'_{ab} in G' which contains a "pushed-forward" bidirected edge guarantees that the above-generated system of equations can be solved for the corresponding variable λ_{ab}.

As per the definition of G', it is identical to G, except:

1. the bidirected edge $x \leftrightarrow y$ is removed
2. $\forall c \in Ch(x)$, the edges $c \leftrightarrow y$ are added.

3. Descendants of y, z, where all edges of δ_{yz} are not known are removed

This new model G', with parameters Λ' and $\mathcal{E}'$ has system of equations:

$$\sigma''_{vw} = \sum_{\pi \in \mathcal{T}_{vw}} \pi(\Lambda', \mathcal{E}') \qquad \forall v, w \in G'$$

We compare this new system of equations to the modified equations of G.

- For all non-descendants of x or y, all covariance equations are identical (Both graphs have the same treks from non-descendants of x and y to all other nodes, and these covariances were not modified in the augmented equations).
- For all descendants of x, the modified equations for G have $\epsilon_{xy}\lambda_{xc}$ wherever G' has ϵ'_{cy} when G does not have bidirected edge $c \leftrightarrow y$. If G already includes an ϵ_{cy}, then it has $(\epsilon_{xy}\lambda_{xc} + \epsilon_{cy})$ for each ϵ'_{cy}. This can be seen by comparing the treks available in the two models. We can create a map of treks in G to treks in G'. Treks not crossing the added/removed bidirected edges are identical. All that remains are treks crossing ϵ_{xy} in G, and ϵ'_{cy} in G'. Suppose we have a trek from a to b in G' $a \leftarrow ... \leftarrow c \leftrightarrow y \rightarrow ... \rightarrow b$, crossing the bidirected edge ϵ'_{cy}. The corresponding trek in G across ϵ_{cy}, if it exists, and the trek $a \leftarrow ... \leftarrow c \rightarrow x \leftrightarrow y \rightarrow ... \rightarrow b$ both map to it. Since we have a map from treks in G to all treks in G', which differs only in the specified spot, the equations are likewise identical save for the mapping difference.
- The covariances between x and the descendants of y and y have likewise identical equations. This is because the removed treks in the modified equations are the only possibilities including ϵ_{xy}, so all variables behave as if the bidirected edge did not exist. This can also be seen by recognizing that setting $\epsilon_{xy} = 0$ would result in the same equation as removing all instances of the variable. Since the only treks from x which include ϵ_{xy} start by crossing $x \leftrightarrow y$, and continue on a directed path, removing all directed paths from y multiplied by ϵ_{xy} achieves the desired effect.

Finally, we notice that any algorithm for identifiability in this new model G' certifies that the system of equations can be uniquely solved for a given parameter, and the answer can be written in terms of Σ''.

We argue that the same parameter can be solved using the augmented equations of the original model G, replacing Σ'' with Σ' in the estimand returned from the identifiability algorithm for G'. This would be directly true if the modified equations for G were really identical to the equations for G'. However, the equations of G differ in ϵ_{xy} and ϵ_{xc} as mentioned above. We show that this difference does not affect the solutions for any of the directed or bidirected edges except ϵ_{xc}.

Looking at the form of the structural equations, we get a sum of treks, which are themselves a product of paths. We exploit the mapping created above between treks in G to treks in G' to get:

$$\sigma''_{wv} = \sum \textit{treks not passing } \epsilon'_{cy} + \sum \textit{treks passing } \epsilon'_{cy}$$

Each trek can pass an ϵ at most once, at the top of the trek. Looking at the treks of G', we get:

$$\begin{aligned}\sum \textit{treks passing } \epsilon'_{cy} &= \delta'_{cw}\epsilon'_{cy}\delta'_{yv} + \delta'_{cv}\epsilon'_{cy}\delta'_{yw} \\ &= (\delta'_{cw} + \delta'_{cv})\epsilon'_{cy}(\delta'_{yv} + \delta'_{yw})\end{aligned}$$

The corresponding equation in G is:

$$\sum \textit{treks passing } \epsilon_{xy} = (\delta_{cw}+\delta_{cv})(\epsilon_{xy}\lambda_{xc}+\epsilon_{cy})(\delta_{yv}+\delta_{yw})$$

With the fact that the δ are all identical in both G and G' in terms of equation structure, we get:

$$\sum \textit{treks passing } \epsilon'_{cy} = (\delta_{cw} + \delta_{cv})\epsilon'_{cy}(\delta_{yv} + \delta_{yw})$$

Our goal now is to replace the $\epsilon_{xy}\lambda_{xc} + \epsilon_{cy}$ from the equation of G with a temporary variable ϵ_{Tc}, giving

$$\sum \textit{treks passing } \epsilon_{xy} = \delta_{cw}\epsilon_{Tc}\delta_{yv}$$

With this new system of equations, the temporary variable ϵ_{Tc} corresponds to ϵ'_{cy}, and the two systems are identical in their unknowns, making any solution in G' a solution for the modified G equations.

To achieve this, we need to show that the substitution is valid. The argument we are making is that if we have a system of equations:

$$\begin{aligned}K_1 &= x + (x + 5) \\ K_2 &= 3x\end{aligned}$$

we can replace $x + 5$ with y, giving:

$$\begin{aligned}K_1 &= x + y \\ K_2 &= 3x \\ y &= x + 5\end{aligned}$$

If the equations

$$K_1 = x + y$$
$$K_2 = 3x$$

are sufficient for uniquely identifying the value of x, then assuming consistency of the original equations, the same solution for x is valid for the full system, including the third equation.

This is exactly the situation we have for our trek equations. We define $\epsilon_{Tc} = \epsilon_{xy}\lambda_{xc} + \epsilon_{cy}$, and if the system of equations excluding the equation constraining ϵ_{Tc} has a solution for a given parameter, the same solution will be valid for the full system.

Lastly, we notice that ϵ_{cy} can be obtained from the solutions to $\epsilon_{Tc}, \epsilon_{xy}, \lambda_{xc}$ using the same equation.

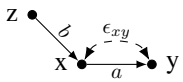
This completes the proof. □

C. Identification, sensitivity analysis and Gröbner bases

Gröbner bases are a symbolic method of computer algebra used to solve systems of polynomial equations. García-Puente et al. (2010) have shown that the identification (ID) problem in linear SCMs can be reduced to solving a system of polynomial equations and how Gröbner bases provide a complete solution.

In this section we will take a practical approach of showing how to set up the ID problem so it can be solved with Gröbner bases. We also show how to extend this to include sensitivity parameters, solving the problem of θ-identification. Our approach is based on García-Puente et al. (2010). For a basic understanding of Gröbner bases, please refer to Cox et al. (1992).

Gröbner bases can be seen as an algorithm to do variable elimination in complex polynomial equations. Let us illustrate the variable elimination approach in the simple instrumental variable graph:



We can write the (normalized) covariance equations induced by the graph as follows:

$$\sigma_{xy} = a + \epsilon_{xy}$$
$$\sigma_{zy} = b \times a$$
$$\sigma_{zx} = b$$

Given these equations, the goal is to solve for a in terms of the covariances of Σ only. Normally, one would approach this directly, by simply eliminating one variable at a time. For example, after eliminating b, we get:

$$\sigma_{xy} = a + \epsilon_{xy}$$
$$\sigma_{zy} = \sigma_{zx} \times a$$

Next, we would eliminate ϵ_{xy}, by putting it in terms of a:

$$\epsilon_{xy} = \sigma_{xy} - a$$

Then, we have a final equation just in terms a and the Σ. This equation can be solved for a, and depending on how many values of a satisfy the constraint, it give us our identification result (here, only $a = \frac{\sigma_{zy}}{\sigma_{zx}}$ is valid).

Gröbner bases perform an equivalent operation—they successively eliminate variables from the system of equations. In this situation, we want to eliminate ϵ_{xy} and b, leaving only a and the covariances. In SAGE (The Sage Developers, 2018), this reduces to the following code:

```
R.<a,b,epsilon_xy,sigma_zx,sigma_zy,sigma_xy>
                                  = PolynomialRing(QQ)
Ideal(
    sigma_xy - (a+epsilon_xy),
    sigma_zy - (b*a),
    sigma_zx - (b)
).elimination_ideal([epsilon_xy,b]).groebner_basis()
```

If the result is a first degree polynomial in a, there is a single solution.

The extension of this method to the θ-identification problem entailed by sensitivity analysis is straightforward. As sensitivity parameters are treated like known variables, we simply do not eliminate them. In the above example, if we were to treat ϵ_{xy} as a sensitivity parameter, our code would be:

```
R.<a,b,epsilon_xy,sigma_zx,sigma_zy,sigma_xy>
                                  = PolynomialRing(QQ)
Ideal(
    sigma_xy - (a+epsilon_xy),
    sigma_zy - (b*a),
    sigma_zx - (b)
).elimination_ideal([b]).groebner_basis()
```

with an identical interpretation: if the resulting polynomials in a, Σ and ϵ_{xy} are linear in a, we conclude that knowing the givens is sufficient to identify a.

Unfortunately, despite the completeness of this approach, Gröbner bases are doubly-exponential in the number of variables, and in this case *each edge* corresponds to a variable (Bardet, 2002). This limits the practical solvable graph size to 4 or 5 nodes (Foygel et al., 2012; García-Puente et al., 2010). Our own experiments hit upon the same limitation, with attempted computations on 5-node graphs sometimes taking several days for identifying single edges, despite using an optimized representation of the equations (Foygel et al., 2012).

D. Detailed description of computational experiments

In this section, we provide a detailed description of our computational experiments, including pseudocode and additional tests. Our computational experiments have two main goals.

First, they aim to empirically verify the generality of our constrained identification algorithm CID, by comparing our results to the ground truth obtained via computer algebra.

Second, note that CID has three separate components:

1. The QID algorithm (Chen et al., 2017), which we use both for the identification of directed edges, and for incorporating constraints on directed edges that can be used as sensitivity parameters;
2. The graphical manipulations performed by PUSHFORWARD, which we use to incorporate constraints on bidirected edges; and,
3. The order in which to perform the graphical manipulation of PUSHFORWARD. In CID we chose to perform a topological ordering as described in Algorithm 1.

Thus, our computational experiments also aim to disentangle the contributions of each of those components to our results.

Solving all 3 and 4-Node sensitivity queries

Our computational experiments rely on the ability to find ground-truth answers to the question of whether a target coefficient λ_{ab} is θ-identifiable in a given graph G (this is defined to be a *query*). As explained in Section C of this supplementary material, these ground truth answers can be obtained with algebraic methods, more precisely using Gröbner bases (García-Puente et al., 2010).

For 3-node models we have 50 connected graphs with 720 possible queries; for 4-node models, we have 3,745 connected graphs and 1,059,156 possible queries. Note that, for 5-node models, we have 1,016,317 connected graphs and 11,615,669,904 possible queries. As mentioned in Section C, ground-truth computations using computer algebra can take hours (or sometimes days) for a *single* 5-node graph, redering an exhaustive study of sensitivity queries in 5-node models impractical.

We have thus performed an exhaustive computation of the ground truth answer of all possible queries in 3 and 4 node models via computer algebra using SAGE (The Sage Developers, 2018). These results give us a list stating for every graph G, every edge λ_{ab}, and *all possible subsets* of directed and bidirected edges used as sensitivity parameters θ, whether λ_{ab} can be uniquely computed from Σ and θ.

Our main interest lies on those queries that can be identified only when $\theta \neq \emptyset$ (we call this a sensitivity query)—in other words, we do not consider those edges that can be identified from Σ alone, since in these cases the parameter is identifiable and a sensitivity analysis would not be needed. The ground truth numbers of all θ-identifiable queries only when $\theta \neq \emptyset$ are 320 for 3-node models and 578,858 for 4-node models.

Our exhaustive computations also allow us to see how many sensitivity queries can be solved using *only subsets of directed edges* or *only subsets of bidirected edges* as sensitivity parameters. The decomposition then becomes the following. For 3-node models, there are 19 sensitivity queries that can be solved using only subsets of directed edges as sensitivity parameters, 109 using only subsets of bidirected edges, and, as before, 320 total queries which are solvable using an arbitrary combination of both. For 4-node models, these numbers increase to 15,740, 52,016 and 578,858 respectively. These numbers reveal that incorporating constraints on bidirected edges is an essential step for deriving sensitivity curves.

Comparing QID and CID to ground-truth answers

Once we have obtained ground-truth answers to all queries in 3 and 4-node models, we run both the QID as well as the CID algorithm for each of those queries and check whether they can correctly decide whether θ is an admissible set of sensitivity parameters for λ_{ab} in G (and thus able to provide a sensitivity curve). This comparison gives us the numbers we have presented in the main text in Table 1.

Alternative ordering methods for PUSHFORWARD

In the main text, the CID algorithm applies PUSHFORWARD in a topological ordering for processing multiple bidirected edges. The method does not perform all possible graphical manipulations, and as such, a valid concern is that it might be less capable than a more general search. Another interesting question is to check whether simpler methods would perform as well as the current CID implementation. To tackle these questions, we tested additional ordering methods for handling multiple bidirected edges.

For simplicity of exposition, the CID algorithm in the main text has the ordering method embedded in the pseudocode itself. For the purposes of this section, however, it is conceptually easier to create a meta algorithm that repeats the following process: (i) first it creates a collection of valid modified graphs $\mathcal{G}$ applying PUSHFORWARD according to some ordering method; then, (ii) it applies an identification algorithm to each of those modified graphs. This is given in Algorithm 3, which we call CID*.

In Algorithm 3, the argument PFORDER represents a func-

			3 NODES			4 NODES		
	PF order	ID Alg. directed edges	*Directed*	*Bidirected*	*Both*	*Directed*	*Bidirected*	*Both*
	none	QID	19	-	68	14,952	-	170,304
	PFO	QID	19	101	304	14,952	43,526	505,076
	PFS	QID	19	105	308	14,952	46,630	517,036
	PFR	QID	19	109	320	14,952	50,708	555,758
(CID)	**PFT**	**QID**	19	109	320	14,952	50,708	555,758
	none	Complete	19	-	68	15,740	-	177,216
	PFO	Complete	19	101	304	15,740	44,680	524,846
	PFS	Complete	19	105	308	15,740	47,962	538,332
	PFR	Complete	19	109	320	15,740	51,992	578,758
	PFT	Complete	19	109	320	15,740	51,992	578,758
	GROUND TRUTH		19	109	320	15,740	52,016	578,858

Table 2: Number of θ-identifiable queries (only when $\theta \neq \emptyset$) per type of sensitivity parameters θ, using different ordering methods for PUSHFORWARD and different ID algorithm for the directed edges. Ground Truth is computed using Gröbner bases. The first column defines the ordering method of PUSHFORWARD used for incorporating constraints on bidirected edges—this is passed as the argument PFORDER in the general function CID*. The second column refers to the identification algorithm used for directed edges—this is passed as the argument IDMETHOD in the general function CID*. "Complete" means we used Gröbner bases to simulate a complete ID algorithm for *directed edges* running inside CID*. Note the first row corresponds to QID and the boldfaced row corresponds to CID as presented in the main text applying PUSHFORWARD in topological ordering. These two rows are the ones presented in Table 1 of the main text. A pseudocode for computing these numbers is given in Algorithm 4.

Algorithm 3 Meta constrained ID algorithm.

function CID*($G, \Sigma, \mathcal{B}, \mathcal{D}$, PFORDER, IDMETHOD)
  repeat
    $\mathcal{G} \leftarrow$ PFORDER$(G, \Sigma, \mathcal{B}, \mathcal{D})$
    for $\langle G', \Sigma' \rangle \in \mathcal{G}$ **do**
      $\mathcal{D} \leftarrow \mathcal{D} \cup$ IDMETHOD$(G', \Sigma', \mathcal{D})$
    end for
  until all directed edges have been identified or no edge has been identified in the last iteration
  return $\mathcal{D}$
end function

tion that takes as inputs a graph G, a covariance matrix Σ, a set of known bidirected edges $\mathcal{B}$ and a set of known directed edges $\mathcal{D}$. It then returns a *collection* $\mathcal{G}$ of valid modified models $\langle G', \Sigma' \rangle$ by iteratively applying PUSHFORWARD following a particular ordering method (for example, topological ordering). The argument IDMETHOD refers to an identification method for directed edges (for instance, QID). It is a function that takes as inputs a graph G, a covariance matrix Σ and a set of known directed edges $\mathcal{D}$ and it returns the new set of known directed edges.

We can now create different functions for different ordering methods. For instance, the function PFT described in Algorithm 7 applies PUSHFORWARD in topological ordering (as embedded in Algorithm 1 of the main text) and returns all valid modified graphs. We now define three additional ordering methods.

- PFO described in Algorithm 5. This function pushes forward each bidirected edge only once, considering the original graph. This method is the simplest application of PUSHFORWARD, and serves as a base of comparison to assess the gains of more elaborate methods.
- PFS described in Algorithm 6. This function tries to apply PUSHFORWARD once to all subsets of bidirected edges connected to each end node. This procedure has exponential computational complexity.
- PFR described in Algorithm 8. This function recursively tries every possible combination of applying PUSHFORWARD for each bidirected edge connected to the same end node (it tries each subset once, and of those that can be pushed forward again, tries each subset, and so on). This procedure has doubly exponential computational complexity.

All these function return a collection $\mathcal{G}$ of valid modified graphs, and can be used as the PFORDER argument in the CID* function. Of these methdos, PFR is arguably the most important for comparison with our current implementation of topological ordering. The results are shown in the first half of Table 2, which compares CID* using the same

ID method for directed edges (QID) but different ordering methods for applying PUSHFORWARD. Our preferred version, which was presented in the main text as CID, corresponds to the boldfaced row with ordering method PFT and ID method QID. As we can see, topological ordering performs as well as the brute-force recursive search of all subsets performed by PFR, which has doubly exponential computational complexity.

Disentangling PF and QID

Finally, the incompleteness of CID can stem from two sources: limitations of the graphical manipulations performed by PUSHFORWARD or the incompleteness of the identification algorithm for directed edges, QID. Separating the two can help guide efforts for future research. To achieve that, we used algebraic methods to simulate how CID would have performed if it had access to a complete identification algorithm for directed edges instead of QID.

More precisely, we use Gröbner bases as our ID algorithm for directed edges (IDMETHOD) in CID*, where, just like QID, Gröbner bases only have access to constraints on bidirected edges via the graphical manipulation performed by PUSHFORWARD. That is, Gröbner bases is dealing with the problem as if it were a "vanilla" identification problem, not explicitly knowing that the bidirected edge is fixed. The results can be seen in the second half of Table 2. The last row indicates, for instance, that incorporating constraints on bidirected edges using PUSHFORWARD in topological order, in combination with a complete identification algorithm for directed edges, would have identified over 99.99% of 4-node sensitivity queries.

This suggests that: (i) the main bottleneck of the current implementation of CID is QID itself; (ii) PUSHFORWARD with topological ordering is an efficient procedure for dealing with bidirected edges that can reap the benefits of improved identification algorithms.

E. The missed cases

As discussed in the previous section, PUSHFORWARD in topological order, in combination with a complete identification algorithm for *directed edges*, would have identified over 99.99% of all 4-node sensitivity queries. In this section we briefly discuss some of the missed cases, which may provide guidance for further improvements of the CID algorithm. We also provide all the missed cases for those interested in exploring them further (Tables 3 and 4).

When iterating over modified graphs, the CID algorithm feeds its next iteration *only* identification results for *direct effects* (single coefficients), not of path specific effects or total effects (sums of products of coefficients), which may nevertheless be identified. Figure 8 shows two simple examples that illustrates how not exploiting the knowledge of known total effects can result in a failure of identification.

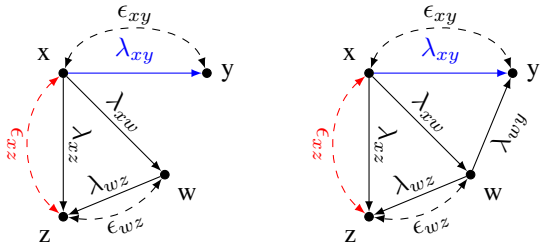


Figure 8: Examples of missed cases using PUSHFORWARD with a complete identification algorithm of *directed edges*. In both examples, λ_{xy} is ϵ_{zx}-identifiable, but the algorithm fails due to lack of exploitation of identified total effects. In example 8a, it turns out a simple marginalization of w suffices for the ϵ_{zx}-identification of λ_{xy} using the current implementation of the CID algorithm. However, marginalization alone is often not enough, as shown in example 8b.

Let us start with Figure 8a. In this example, our task is to find a sensitivity curve for λ_{xy} in terms of ϵ_{zx}. First note that z is not a valid instrument for λ_{xy} since it is a descendant of x. However, pushing forward ϵ_{zx} allows us to identify the *total effect* of x on z. This, in turn, permits the creation of the auxiliary variable $z^* = z-(\lambda_{xz}+\lambda_{xw}\lambda_{wz})x$ which is now a valid instrument for λ_{xy}. In the example of Figure 8a, it turns out a simpler solution would also suffice—marginalizing w. Note the marginalized DAG results in a simple three node model which can be solved by the current implementation of CID. Nevertheless, marginalization by itself may not always be sufficient, as a simple variation of this very example shows (Figure 8b).

In sum, θ-identification in these cases require systematically exploiting known total effects (for instance, creating AVs subtracting out total effects) or known path-specific effects,

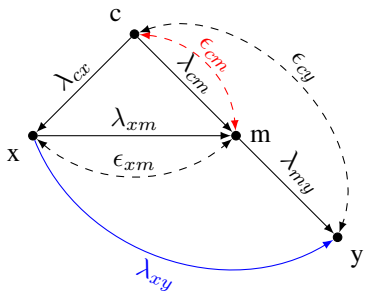
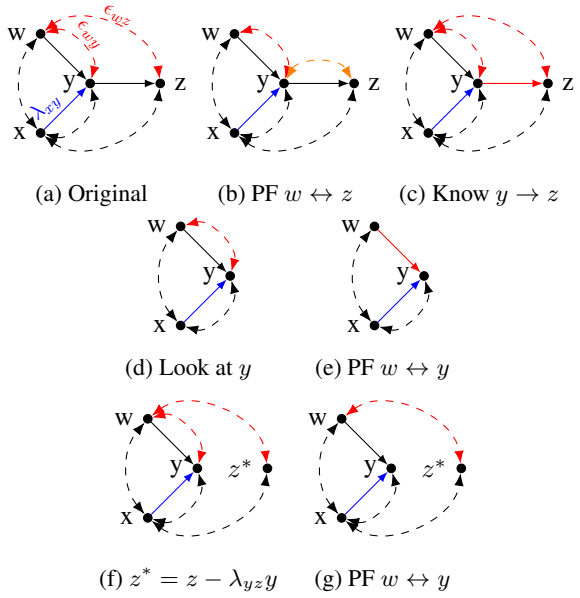


Figure 9: An interesting missed case example. Here λ_{xy} is ϵ_{cm}-identifiable. All examples can be found in Tables 3 and 4.

a task which still does not have a satisfactory solution in the literature. A final interesting (and challenging) example in which CID failed to find the sensitivity curve is shown in Fig .9.

F. Utility of descendants

Here we show that not pruning descendants of variable y can be useful for identification. An example is given in Fig. 10.



(a) Original (b) PF $w \leftrightarrow z$ (c) Know $y \to z$

(d) Look at y (e) PF $w \leftrightarrow y$

(f) $z^* = z - \lambda_{yz} y$ (g) PF $w \leftrightarrow y$

Figure 10: Take the graph in 10a. The red bidirected edges $(\epsilon_{wy}, \epsilon_{wz})$ are assumed to be known, and the target quantity is λ_{xy} (blue). First, we can use knowledge of $w \leftrightarrow z$ to use w as an instrument for $y \to z$ (10b,10c). This knowledge, however, does not help in solving for $x \to y$ (10d), even when pushing forward $w \leftrightarrow y$ (10e). The issue is that we pruned z when pushing forward $w \leftrightarrow y$, since it is a descendant of y. However, note we can create an AV z^*, which behaves as if it were not a descendant of y (10f). It turns out that z^* is an instrument for $x \to y$ conditioned on w in 10g, solving the problem!

Algorithm 4 Pseudocode for checking performance of CID* with different PUSHFORWARD orders and different ID algorithm for directed edges. In the code, IDENTIFYDIRECTEDEDGES and ISIDENTIFIED are computed using computer algebra (Gröbner bases), and give the ground-truth values.

initialize Total $\leftarrow 0$
initialize PFtotal $\leftarrow 0$
$\mathcal{S} \leftarrow$ set of all possible connected DAGs, with all combinations of directed and bidirected edges.
for each graph $\langle G, \Sigma \rangle \in \mathcal{S}$ **do**
  IDedges $\leftarrow$ IDENTIFYDIRECTEDEDGES$(\mathcal{G})$
  for all $(x \to y) \in \mathcal{G}$ **where** $(x \to y) \notin$ IDedges **do**
    $\mathcal{SPS} \leftarrow$ All subsets of directed and bidirected edges of G which do not contain $(x \to y)$
    for each set $\langle \mathcal{D}, \mathcal{B} \rangle \in \mathcal{SPS}$ **do**
      if ISIDENTIFIED$(\mathcal{G}, x \to y, \mathcal{D}, \mathcal{B})$ **then**
        Total $\leftarrow$ Total $+ 1$
        if $(x \to y) \in$ CID*$(G, \Sigma, \mathcal{D}, \mathcal{B},$ PFORDER, IDMETHOD$)$ **then**
          PFTotal $\leftarrow$ PFTotal $+ 1$
        end if
      end if
    end for
  end for
end for
return $\frac{PFtotal}{Total}$

Algorithm 5 Push forward each bidirected edge once.

function PFO$(G, \Sigma, \mathcal{B}, \mathcal{D})$
  let $\mathcal{B}_y$ represent subset of $\mathcal{B}$ where all edges have y as end point $(B_y = \{(x \leftrightarrow y) \in \mathcal{B}, \forall x\})$
  $\mathcal{G} \leftarrow \{(G, \Sigma)\}$
  for each node $y \in G$ **do**
    for bidirected edge $\epsilon_{xy} \in \mathcal{B}_y$ **do**
      if $x \notin$ DE(y) **or** $\delta_{yx} \in \mathcal{D}$ **then**
        add PF$(G, \Sigma, \mathcal{D}, \epsilon_{xy}, x)$ **to** $\mathcal{G}$
      end if
    end for
  end for
  return $\mathcal{G}$
end function

Algorithm 6 Push forward all subsets once.

function PFS($G, \Sigma, \mathcal{B}, \mathcal{D}$)
  let $\mathcal{B}_y$ represent subset of $\mathcal{B}$ where all edges have y as end point ($B_y = \{(x \leftrightarrow y) \in \mathcal{B}, \forall x\}$)
  $\mathcal{G} \leftarrow \{(G, \Sigma)\}$
  for each node y **do**
    for each $B'_y \subseteq B_y$ **do**
      $\langle G', \Sigma' \rangle \leftarrow \langle G, \Sigma \rangle$
      for each $\epsilon_{xy} \in B'_y$ **do**
        if $x \notin \text{DE}(y)$ **or** $\delta_{yx} \in \mathcal{D}$ **then**
          $\langle G', \Sigma' \rangle \leftarrow$PF($G', \Sigma', \mathcal{D}, \epsilon_{xy}, x$)
          for all $z \in \text{CH}(x)$ **do**
            if $\lambda_{xz} \notin \mathcal{D}$ **then**
              remove ϵ_{zy} from $\mathcal{B}'_y$ if it was not yet processed.
            end if
          end for
        end if
      end for
      add $\langle G', \Sigma' \rangle$ **to** $\mathcal{G}$
    end for
  end for
  return $\mathcal{G}$
end function

Algorithm 7 Push forward in topological order.

function PFT($G, \Sigma, \mathcal{B}, \mathcal{D}$)
  let $\mathcal{B}_y$ represent subset of $\mathcal{B}$ where all edges have y as end point ($B_y = \{(x \leftrightarrow y) \in \mathcal{B}, \forall x\}$)
  $\mathcal{G} \leftarrow \{(G, \Sigma)\}$
  for each node y **do**
    $\langle G', \Sigma' \rangle \leftarrow \langle G, \Sigma \rangle$
    for each $\epsilon_{xy} \in B_y$ in topological order on x **do**
      if $x \notin \text{DE}(y)$ **or** $\delta_{yx} \in \mathcal{D}$ **then**
        $\langle G', \Sigma' \rangle \leftarrow$PF($G', \Sigma', \mathcal{D}, \epsilon_{xy}, x$)
        add $\langle G', \Sigma' \rangle$ **to** $\mathcal{G}$
        for all $z \in \text{CH}(x)$ **do**
          if $\lambda_{xz} \notin \mathcal{D}$ **then**
            remove ϵ_{zy} from $\mathcal{B}_y$ if it was not yet processed.
          else
            add ϵ_{zy} to $\mathcal{B}_y$
          end if
        end for
      end if
    end for
  end for
  return $\mathcal{G} \cup$ PFO($G, \Sigma, \mathcal{B}, \mathcal{D}$)
end function

Algorithm 8 Push forward all subsets recursively.

function PFR($G, \Sigma, \mathcal{B}, \mathcal{D}$)
  let $\mathcal{B}_y$ represent subset of $\mathcal{B}$ where all edges have y as end point ($B_y = \{(x \leftrightarrow y) \in \mathcal{B}, \forall x\}$)
  initialize $\mathcal{G} \leftarrow \{(G, \Sigma, \emptyset)\}$
  for each node y **do**
    $PushSets \leftarrow \{\langle G, \Sigma, B'_y \rangle$ for all $B'_y \subseteq B_y\}$
    while $PushSets$ **not empty do**
      pop $\langle G', \Sigma', B'_y \rangle$ from $PushSets$
      $PushAgain \leftarrow \{\}$
      for each $\epsilon_{xy} \in B'_y$ **do**
        if $x \notin \text{DE}(y)$ **or** $\delta_{yx} \in \mathcal{D}$ **then**
          $\langle G', \Sigma' \rangle \leftarrow$PF($G', \Sigma', \mathcal{D}, \epsilon_{xy}, x$)
          for all $z \in \text{CH}(x)$ **do**
            if $\lambda_{xz} \notin \mathcal{D}$ **then**
              remove ϵ_{zy} from $\mathcal{B}'_y$ if it was not yet processed.
            else
              add ϵ_{zy} **to** $PushAgain$
            end if
          end for
        end if
      end for
      add $\langle G', \Sigma' \rangle$ **to** $\mathcal{G}$
      for all$B''_y \subseteq PushAgain$ **do**
        add $\langle G', \Sigma', B''_y \rangle$ **to** $PushSets$
      end for
    end while
  end for
  return $\mathcal{G}$
end function

	Graph	Target Quantity	Sensitivity Parameters
1	1→2 1→3 1→4 2→4 1↔3 1↔4 2↔4	1→3	1↔4
2	1→2 1→3 1→4 2→4 1↔3 1↔4 2↔4	1→3	1↔4 1→2
3	1→2 1→3 1→4 2→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4
4	1→2 1→3 1→4 2→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 1→2
5	1→2 1→3 1→4 2→4 3→4 1↔3 1↔4 2↔4	1→3	1↔4 3→4
6	1→2 1→3 1→4 2→4 3→4 1↔3 1↔4 2↔4	1→3	1↔4 1→2 3→4
7	1→2 1→3 1→4 2→4 3→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 3→4
8	1→2 1→3 1→4 2→4 3→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 1→2 3→4
9	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4	1→3	1↔4
10	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4	1→3	1↔4 2→3
11	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4	1→3	1↔4 1→2
12	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4	1→3	1↔4 1→2 2→3
13	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4
14	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 2→3
15	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 1→2
16	1→2 1→3 2→3 1→4 2→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 1→2 2→3
17	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4	1→3	1↔4 3→4
18	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4	1→3	1↔4 3→4 1→2
19	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4	1→3	1↔4 2→3 3→4
20	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4	1→3	1↔4 2→3 3→4 1→2
21	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 3→4
22	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 3→4 1→2
23	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 2→3 3→4
24	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 1↔4 2↔4 3↔4	1→3	1↔4 3↔4 2→3 3→4 1→2
25	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4	2→4	1↔3
26	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4	2→4	1↔3 1→2
27	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4	3→4	1↔3
28	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4	3→4	1↔3 1→2
29	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4 3↔4	2→4	1↔3 3↔4
30	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4 3↔4	2→4	1↔3 3↔4 1→2
31	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4 3↔4	3→4	1↔3 3↔4
32	1→2 1→3 2→3 2→4 3→4 1↔3 2↔3 1↔4 3↔4	3→4	1↔3 3↔4 1→2
33	1→2 1→3 2→3 1→4 1↔3 2↔3 1↔4	1→4	1↔3
34	1→2 1→3 2→3 1→4 1↔3 2↔3 1↔4	1→4	1↔3 1→2
35	1→2 1→3 2→3 1→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4
36	1→2 1→3 2→3 1→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 1→2
37	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3
38	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 3→4
39	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 1→2
40	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 1→2 3→4
41	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4
42	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 3→4
43	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 1→2
44	1→2 1→3 2→3 1→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 1→2 3→4
45	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4	1→4	1↔3
46	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4	1→4	1↔3 2→4
47	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4	1→4	1↔3 1→2
48	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4	1→4	1↔3 1→2 2→4
49	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4
50	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 2→4

Table 3: Missed sensitivity queries of PUSHFORWARD in topological order, in combination with a complete identification algorithm for *directed edges*. Part 1.

	Graph	Target Quantity	Sensitivity Parameters
51	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 1→2
52	1→2 1→3 2→3 1→4 2→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 1→2 2→4
53	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 3→4
54	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 3→4 1→2
55	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 2→4
56	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 1→2 2→4
57	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 3→4 2→4
58	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 3→4 1→2 2→4
59	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 1↔4
60	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	1→4	1↔3 1↔4 1→2
61	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	2→4	1↔3 1→4
62	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	2→4	1↔3 1→2 1→4
63	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	2→4	1↔3 1↔4
64	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	2→4	1↔3 1↔4 1→2
65	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	3→4	1↔3 1→4
66	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	3→4	1↔3 1→2 1→4
67	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	3→4	1↔3 1↔4
68	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4	3→4	1↔3 1↔4 1→2
69	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 3→4
70	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 3→4 1→2
71	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 2→4
72	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 1→2 2→4
73	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 3→4 2→4
74	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 3↔4 3→4 1→2 2→4
75	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 1↔4 3↔4
76	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	1→4	1↔3 1↔4 3↔4 1→2
77	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	2→4	1↔3 3↔4 1→4
78	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	2→4	1↔3 3↔4 1→2 1→4
79	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	2→4	1↔3 1↔4 3↔4
80	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	2→4	1↔3 1↔4 3↔4 1→2
81	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	3→4	1↔3 3↔4 1→4
82	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	3→4	1↔3 3↔4 1→2 1→4
83	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	3→4	1↔3 1↔4 3↔4
84	1→2 1→3 2→3 1→4 2→4 3→4 1↔3 2↔3 1↔4 3↔4	3→4	1↔3 1↔4 3↔4 1→2
85	1→2 2→3 2→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4
86	1→2 2→3 2→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4 2→3
87	1→2 2→3 2→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4
88	1→2 2→3 2→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4 2→3
89	1→2 2→3 1→4 2→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4 1→4
90	1→2 2→3 1→4 2→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4 2→3 1→4
91	1→2 2→3 1→4 2→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4 1→4
92	1→2 2→3 1→4 2→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4 2→3 1→4
93	1→2 1→3 1→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4
94	1→2 1→3 1→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4 1→3
95	1→2 1→3 1→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4
96	1→2 1→3 1→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4 1→3
97	1→2 1→3 1→4 2→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4 2→4
98	1→2 1→3 1→4 2→4 3→4 1↔2 1↔4 3↔4	1→2	1↔4 1→3 2→4
99	1→2 1→3 1→4 2→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4 2→4
100	1→2 1→3 1→4 2→4 3→4 1↔2 1↔4 2↔4 3↔4	1→2	1↔4 2↔4 1→3 2→4

Table 4: Missed sensitivity queries of PUSHFORWARD in topological order, in combination with a complete identification algorithm for *directed edges*. Part 2.