Consistent Estimation given Data that are Missing Not At Random

Karthika Mohan KARTHIKA@CS.UCLA.EDU

Judea Pearl JUDEA@CS.UCLA.EDU

Abstract

This paper presents a unified approach for recovering causal and probabilistic queries using graphical models given missing (or incomplete) data. To this end, we develop a general algorithm that can recover conditional probability distributions and conditional causal effects in semi-Markovian models.

Keywords: Missing Not At Random; Missing Data; Causal Bayesian Network.

1. Introduction

Missing data or incomplete data are data in which values of one or more variables are unobserved/not available. Missing data could arise due to multiple reasons: reluctance of people to reveal sensitive information such as salary, tax reports and heath related information, badly designed questionnaires and accidental deletion of records.

As an example, consider data collected over the following variables: Education, Gender, Salary, Age and University attended, in which Salary and Age are corrupted by missingness. We also know that respondents with very high and very low salary were reluctant to reveal their salaries and due to a random printing error, a few questionnaires did not have the question pertaining to age printed on them. Given the incomplete data and the graphical model encoding the underlying assumptions about the data generation process, we are interested in answering questions such as: (i) what is the effect of age and gender on salary? (ii) what is the effect of gender on education? and (iii) how likely is it to get a salary above $100,000 if you attended Cornell University?

Previous work in consistently estimating (i.e. recovering) causal and probabilistic relations using graphical models include (Daniel et al., 2012; Mohan et al., 2013; Mohan and Pearl, 2014; Shpitser et al., 2015; Shpitser, 2016). That said, missing data is a widely studied problem afflicting all branches of empirical sciences and has resulted in a rich body of literature in diverse fields such as statistics, machine learning, epidemiology and psychology (Rubin, 1996; Little and Rubin, 2002; Enders, 2010; Graham, 2012; Darwiche, 2009; Koller and Friedman, 2009; Robins, 1997; Gill and Robins, 1997)). However in this work we focus exclusively on consistently estimating causal and probabilistic distributions given a graphical model encoding the missingness assumptions.

1.1 Missingness Graphs

Let $G(\mathbf{V}, E)$ be the causal DAG where $\mathbf{V}$ is the set of nodes and E is the set of edges. Nodes in the graph correspond to variables in the dataset and are partitioned into five categories, i.e.

$$\mathbf{V} = V_o \cup V_m \cup U \cup V^* \cup R$$

V_o is the set of variables that are observed in all records in the population and V_m is the set of variables that are missing in at least one record. Variable X is termed as *fully observed* if $X \in V_o$ and *partially observed* if $X \in V_m$. R_{v_i} and V_i^* are two variables associated with every partially

observed variable, where V_i^* is a proxy variable that is actually observed, and R_{v_i} represents the status of the causal mechanism responsible for the missingness of V_i^*; formally,

$$v_i^* = f(r_{v_i}, v_i) = \begin{cases} v_i & \text{if } r_{v_i} = 0 \\ m & \text{if } r_{v_i} = 1 \end{cases} \quad (1)$$

V^* is the set of all proxy variables and $\mathbf{R}$ is the set of all causal mechanisms that are responsible for missingness. Unless stated otherwise it is assumed that no variable in $V_o \cup V_m \cup U$ is a child of an R variable. U is the set of unobserved nodes, also called latent variables. Two nodes X and Y can be connected by a directed edge i.e. $X \rightarrow Y$, indicating that X is a cause of Y, or by a bi-directed edge $X <- > Y$ denoting the existence of a U variable that is a parent of both X and Y.

This graphical representation is called a *Missingness Graph* (or m-graph) (Mohan et al., 2013). Figure 1 (a) exemplifies an m-graph in which $V_o = \{Y\}$, $V_m = \{X\}$, $V^* = \{X^*\}$, $U = \emptyset$ and $R = \{R_X\}$. For any set of variables Z, $Z_m = Z \cap V_m$. Proxy variables may not always be explicitly shown in m-graphs in order to keep the figures simple and clear. The missing data distribution, $P(V^*, V_o, R)$ is referred to as the *manifest distribution* and the distribution that we would have obtained had there been no missingness, $P(V_o, V_m, R)$ is called the *underlying distribution*. $G_{\overline{X}}$ denotes the graph obtained by deleting from G all arrows pointing to nodes in X. Conditional Independencies are read off the graph using the d-separation criterion (Pearl, 2009).

1.2 Types of Missingness

(Rubin, 1976) classified missing data into three categories: Missing Completely At Random (MCAR), Missing At Random (MAR) and Missing Not At Random (MNAR) based on the statistical dependencies between the missingness mechanisms (R variables) and the variables in the dataset (V_m, V_o). The graph based definition of these categories as stated in (Mohan et al., 2013) is: Data are MCAR if $V_m \cup V_o \cup U \perp\!\!\!\perp R$ holds in the m-graph, MAR if $V_m \cup U \perp\!\!\!\perp R | V_o$ holds, and MNAR otherwise.

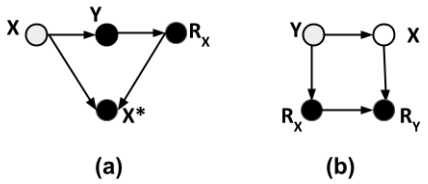

Figure 1: m-graphs in which (a) $P(X, Y)$ is recoverable, (b) $P(X|Y)$ is recoverable.

1.3 Recoverability

Definition 1 (Recoverability of target quantity Q (Mohan et al., 2013)) *Let A denote the set of assumptions about the data generation process and let Q be any functional of the underlying distribution $P(V_m, V_O, R)$. Q is recoverable if there exists an algorithm that computes a consistent estimate of Q for all strictly positive manifest distributions $P(V^*, V_o, R)$ that may be generated under A.*

Since we encode all assumptions in the structure of the m-graph G, recoverability becomes a property of the pair $\{Q, G\}$, and not of the data.

X^*	Y	R_X	$P(X^*, Y, R_X)$
0	0	0	0.2
0	1	0	0.2
1	0	0	0.1
1	1	0	0.1
m	0	1	0.2
m	1	1	0.2

Table 1: Manifest distribution corresponding to m-graph in figure 1(a)

X	Y	$P(X,Y)$
0	0	$\frac{0.2}{0.3} * 0.5 = \frac{1}{3}$
0	1	$\frac{0.2}{0.3} * 0.5 = \frac{1}{3}$
1	0	$\frac{0.1}{0.3} * 0.5 = \frac{0.5}{3}$
1	1	$\frac{0.1}{0.3} * 0.5 = \frac{0.5}{3}$

Table 2: Recovered distribution $P(X, Y)$

Example 1 *Given the m-graph G in figure 1(a) and data in table 1 we will detail the procedure to recover $P(X, Y)$.*
$P(X, Y) = P(X|Y)P(Y)$. *Since $X \perp\!\!\!\perp R_x|Y$ in G we have, $P(X, Y) = P(X|Y, R_x = 0)P(Y)$. It follows from equation 1 that when $R_x = 0$, $X = X^*$. Therefore, $P(X, Y) = P(X^*|Y, R_x = 0)P(Y)$. Notice that $P(X, Y)$ has been expressed in the terms of observed data i.e. $P(X, Y)$ has been recovered. The recovered distribution is given in table 2.*

1.4 Inducing Path

Definition 2 (Inducing Path (Verma and Pearl, 1991)) *A path p between nodes A and B is called an inducing path if all the intermediate nodes on p are colliders and ancestors of A and/or B.*

In the presence of such an inducing path, $\not\exists C$ such that $A \perp\!\!\!\perp B|C$. In this paper we detail how $P(X)$ can be recovered when R_x and X are connected by inducing path(s). Functions *detectIP* and *handleIP* in the appendix detect problematic inducting paths and apply rules 2 and 3 of do-calculus to transform the query so that in the corresponding mutilated graph inducing paths cease to exist. We formalize a general procedure for recoverability in the following sections.

2. Unified Approach to Recoverability of Causal and Probabilistic Queries: An Overview

Results in (Mohan and Pearl, 2014) confirm the need for procedures recovering conditional causal effects to be able to handle conditional probability distributions. Similarly, results in (Shpitser et al., 2015), show that in order to establish recoverability, especially in models with inducing paths, one might need to convert a probablistic query into a causal one using rules of do calculus (Pearl, 2009). Hence we propose a unified approach for handling both causal and probabilistic queries. Our approach relies on the notion of **partial-recoverability**, defined below.

Definition 3 (Partial Recoverability of $P(A|B, \hat{c})$) *Let $D = (A \cup B) \cap V_m$. A query of the form $P(A|B, \hat{c})$ is said to be partially-recovered if $D = \emptyset$ or $R_D \subseteq C$.*

In words, A and B must either not contain partially observed variables or if they do, the corresponding R variables should be present in C. Given a query $P(X|Y, do(D))$ where $\{X, Y, D\} \subseteq V_m \cup V_o$,

we first convert it into partially recovered form and then apply techniques to recover the causal effect. The function isParRec in the appendix checks if an input query is in partially recovered form. The following subsection presents the factorization scheme that we apply to these problems.

2.1 General Missingness Factorization for Recovering $P(X|Y, \hat{w})$

Definition 4 (General Missingness Factorization) *Let X, Y and W be disjoint subsets of variables. Let $D \subseteq X_m \cup Y_m$ be a maximal set such that $\forall D_1 \in D$, $R_{D_1} \notin X \cup Y \cup W$. The factorization:*

$$P(X|Y, \hat{w}) = \frac{P(X|Y, R_D = 0, \hat{w})P(R_D = 0|Y, \hat{w})}{P(R_D = 0|X, Y, \hat{w})} \tag{2}$$

is called General Missingness Factorization (GMF).

$P(X|Y, \hat{z}) = \frac{P(X|Y,R_{xy}=0,\hat{z})P(R_{xy}=0|Y,\hat{z})}{P(R_{xy}=0|X,Y,\hat{z})}$ We used $R_{xy} = 0$ as a shorthand for $R_x = 0, R_y = 0$ above. Let $R_{D_1} < R_{D_2} < ... < R_{D_n}$ denote a (reverse) topological ordering of variables in set R_D such that all child nodes are ordered before their respective parent nodes. $R_D^{(i)}$ denotes the set $\{R_{D_1}, R_{D_2}, ..., R_{D_i}\}$ and $R_D^{(0)} = \emptyset$. $P(R_D = 0|Y, \hat{w})$ and $P(R_D = 0|X, Y, \hat{w})$ in equation (2) can now be factorized as:

$$P(R_D = 0|Y, \hat{w}) = \prod_{R_{D_i} \in R_D} P(R_{D_i} = 0|Y, R_D^{(i-1)} = 0, \hat{w}) \tag{3}$$

$$P(R_D = 0|X, Y, \hat{w}) = \prod_{R_{D_i} \in R_D} P(R_{D_i} = 0|X, Y, R_D^{(i-1)} = 0, \hat{w}) \tag{4}$$

Let $Z_i = \{Y, R_D^{(i-1)}\}$. Then the factors in the numerator and denominator in equation (2) can be compactly denoted as $P(R_{D_i} = 0|Z_i)$ and $P(R_{D_i} = 0|Z_i, X)$, respectively.

Lemma 1 *If $R_{D_i} \perp\!\!\!\perp X|Z_i$ in $G_{\overline{W}}$, equation (2) can be simplified by removing from it both $P(R_{D_i} = 0|Z_i, \hat{w})$ and $P(R_{D_i} = 0|X, Z_i, \hat{w})$.*

Observe that whenever $Y \neq \emptyset$ in $P(X|Y, \hat{w})$ there exists a term $P(R_{D_i} = 0|Z_i, \hat{w})$ in the numerator corresponding to every term $P(R_{D_i} = 0|Z_i, X, \hat{w})$ in the denominator. The following lemma describes how to recover $P(R_{D_i} = 0|Z_i, \hat{w})$ when $P(R_{D_i} = 0|Z_i, X, \hat{w})$ is recoverable.

Lemma 2 ***Recoverability of** $P(R_{D_i} = 0|Z_i, \hat{w})$ **from** $P(R_{D_i} = 0|Z_i, X, \hat{w})$ If $P(R_{D_i} = 0, Z_i|\hat{w})$ and $\forall X = x$, $P(R_{D_i} = 0|X, Z_i, \hat{w})$ and $P(R_{D_i} = 0, X, Z_i|\hat{w})$ are recoverable, then $P(R_{D_i} = 0|Z_i, \hat{w})$ can be recovered as, $\frac{P(R_{D_i}=0,Z_i|\hat{w})}{\sum_X \frac{P(R_{D_i}=0,X,Z_i|\hat{w})}{P(R_{D_i}=0|X,Z_i,\hat{w})}}$.*

When $W = \emptyset$, partial recoverability implies recoverability, as exemplified below.

Example 2 *Consider the problem of recovering $P(X|Y)$ given the m-graph in figure 1 (b). We proceed by applying GMF which yields:*

$$\begin{aligned} P(X|Y) &= \frac{P(X|Y, R_{xy} = 0)P(R_y = 0|Y, R_x = 0)P(R_x = 0|Y)}{P(R_y = 0|X, Y, R_x = 0)P(R_x = 0|X, Y)} \\ &= \frac{P(X|Y, R_x = 0, R_y = 0)P(R_y = 0|Y, R_x = 0)}{P(R_y = 0|X, Y, R_x = 0)} \text{ (Applying lemma 1)} \end{aligned}$$

$P(X|Y)$ is recoverable if all factors in the preceding equation are recoverable. $P(X|Y, R_x = 0, R_y = 0) = P(X^*|Y^*, R_x = 0, R_y = 0)$, and hence is recoverable. Since $R_x \perp\!\!\!\perp X|(Y, R_y)$, $P(R_y = 0|X, Y, R_x = 0)$ can be recovered as $P(R_y = 0|X^*, R_x = 0)$. Recoverability of $P(X|Y)$ now hinges on the recoverability of $P(R_y = 0|Y, R_x = 0)$. Given the following, recoverability of $P(R_y = 0|Y, R_x = 0)$ follows from lemma 2.

- $P(R_y = 0, Y, R_y = 0) = P(R_y = 0, Y^*, R_y = 0)$ *(using eq 1)*
- $P(R_y = 0|X, Y, R_x = 0) = P(R_y = 0|X^*, R_x = 0)$, $\forall X = x$ *(using $R_y \perp\!\!\!\perp Y|X, R_x$, eq 1)*
- $P(R_y = 0, X, Y, R_x = 0) = P(R_y = 0, X^*, Y^*, R_x = 0)$ $\forall X = x$ *(using eq 1)*

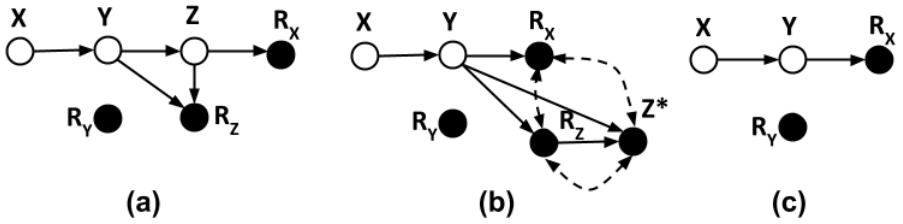

Figure 2: Intermediate graphs corresponding to Example 3

Algorithm 1:
recoverAll($\mathbf{P}(\mathbf{X}|\mathbf{Y}, \hat{\mathbf{d}}), \mathbf{G}, \mathbf{H}, \mathbf{addVar}$)

1: $Y \leftarrow$ pruneQ($\mathbf{P}(\mathbf{X}|\mathbf{Y}, \hat{\mathbf{d}}), \mathbf{G}$), $S \leftarrow \emptyset$
2: $G \leftarrow$ getAncGraph($\mathbf{X} \cup \mathbf{Y} \cup \mathbf{D}, \mathbf{G}$)
3: **If** addVar **then** $S \leftarrow$ addVars($\mathbf{P}(\mathbf{X}|\mathbf{Y}, \hat{\mathbf{d}}), \mathbf{G}$)
4: (H, flag)$\leftarrow$ seen($H, P(X, S|Y, \hat{d}), G$)
5: **If** flag **then return** FAIL
6: $E \leftarrow$ isParRec($P(X, S|Y), G$)
7: **If** $E \neq$ FAIL **then**
8: **If** $D == \emptyset$ **then return** $\sum_S E$
9: $E \leftarrow recoverCausal(E, G, H)$
10: **If** $E \neq FAIL$ **then return** $\sum_S E$
11: $(F_3, F_1, F_2) \leftarrow$ gmf($P(X \cup S|Y), G$)
12: $E \leftarrow$ recFactors($P(X|Y, \hat{d}), S, F_3, F_1, F_2, G, H$)
13: **If** $E == FAIL$ **then return** FAIL
14: **return** $\sum_S E$

Function:
recFactors($\mathbf{P}(\mathbf{X}|\mathbf{Y}, \hat{\mathbf{d}}), \mathbf{S}, \mathbf{F_3}, \mathbf{F_1}, \mathbf{F_2}, \mathbf{G}, \mathbf{H}$)

1: $F_3 \leftarrow$ recoverAll($F_3, G, H, false$)
2: **If** $F_3 == FAIL$ **then return** FAIL
3: **If** $F_2 == \emptyset$ **then return** F_3
4: $(F_2, Latent, flag) \leftarrow recover\_Dr(P(X|Y), S, F_2, G, H)$
5: **If** flag **then return** F_2
6: **If** $F_2 == FAIL$ **then**
7: **If** $Latent == \emptyset$ **then return** $FAIL$,
8: $G^* \leftarrow getLatentGraph(Latent, G)$
9: **return** recoverAll($P(X, S - Latent|Y, \hat{d}), G^*, H, false$)
10: **If** $F_1 == \emptyset$ **then return** $\frac{F_3}{\prod_j F_2[j]}$
11: $(F_1, Latent) \leftarrow recover\_Nr(P(X|Y), S, F_1, G, H)$
12: **If** $F_1 == FAIL$ **then**
13: **If** $Latent == \emptyset$ **then return** FAIL
14: $G^* \leftarrow getLatentGraph(Latent, G)$
15: **return** recoverAll($P(X, S - Latent|Y, \hat{d}), G^*, H, false$)
16: **return** $\frac{F_3 * \prod_i F_1[i]}{\prod_j F_2[j]}$

3. Algorithm: RecoverAll

RecoverAll is the main result of this work and it takes the following inputs: query, m-graph, history tracking variable H (initialized to $\emptyset$) and addVar (initialized to true). We will exemplify the workings of this algorithm using two examples.

Example 3 We demonstrate recoverability using algorithm 1 when given the following inputs: $P(X)$, m-graph G in figure 2 (a), history variable $H = \emptyset$ and addVar initialized to *true*.
Execution of Algorithm RecoverAll:

1. Since $P(X)$ is not a conditional distribution, function $pruneQ$ returns $\emptyset$. $S \leftarrow \emptyset$.
2. The function $getAncGraph$ prunes the graph by recursively removing nodes belonging to $V_m \cup V_o$ that are not pertinent to the recovery procedure. In this case, all such nodes are on the path between X and R_X, hence none are removed and G remains unchanged.
3. $addVars$ identifies the variables to be included in the analysis taking care not to add variables (such as colliders and their descendants) that can unblock paths. Since addVar is true, addVars returns $S = \{Y, Z\}$.
4. $seen$ checks if the input query-graph pair has been already processed in the current context (call stack). Its purpose is to prevent infinite recursion of the algorithm. It adds $(P(X, Y, Z), G)$ to H and returns $(H, false)$.
5. Since flag=false, we move to the next step.
6. $isParRec$ checks if $P(X)$ is in its partially recovered form or if it can be converted into its partially recovered form using eq 1. It returns fail.
11. Function gmf returns $(F_3 = P(X, Y, Z, R_x = 0, R_y = 0, R_z = 0), F_1 = \emptyset, F_2 = \{P(R_x = 0|X, Y, Z, R_z = 0, R_y = 0), P(R_z = 0|X, Y, Z, R_y = 0), P(R_y = 0|X, Y, Z)\}$.
12. Function $recFactors$ is now invoked. $\boxed{A}$
Execution of Function recFactors

1. $recoverAll$ is invoked to recover F_3. As before $Y = \emptyset$ and G remains unchanged. Since addVar is false, $S = \emptyset$. In step 4, (F_3, G) (fig 2(a)) is appended to H. $isPartiallyRecovered$ returns $P(X^*, Y^*, Z^*, R_x = 0, R_y = 0, R_z = 0)$. In step 8, since F_3 is not a causal query, $P(X^*, Y^*, Z^*, R_x = 0, R_y = 0, R_z = 0)$ is returned.
2. Since F_3 is not equal to FAIL, we proceed to step 3.
3. Since $F_2 \neq \emptyset$, $recover\_Dr$ function is invoked. $\boxed{B}$
Execution of Function recover_Dr

2. Each factor in F_2 is processed sequentially in step 2.
3. $P(R_y = 0|X, Y, Z)$ is pruned to yield $P(R_y = 0)$.
4. Graph is updated in the next step by making Y a latent variable.
5. No inducing paths are detected and we move to step 9.
9. *recoverAll* is invoked to recover $P(R_y = 0)$. Step 2 in *recoverAll* reduces the graph to the single node R_y. Step 8, returns the estimand as $P(R_y = 0)$. Control passes back to *recover_Dr*.
10. Success is updated: $Success \leftarrow \{P(R_y = 0)\}$.
Control passes back to step 2 to process the next factor $P(R_x = 0|X, Y, Z, R_z = 0, R_y = 0)$. It is pruned to yield $P(R_x = 0|Z)$. As in the previous case *recoverAll* is invoked in step 9. Step 8 in *recoverAll*, returns the estimand as $P(R_x = 0|Z^*, R_z = 0)$. In line 10 of *recover_Dr*, success is updated: $Success \leftarrow \{P(R_y = 0), P(R_x = 0|Z^*, R_z = 0)\}$. Again control passes back to step 2 to

process the last factor $P(R_z = 0|X, Y, Z, R_y = 0)$. It is pruned to yield $P(R_z = 0|Y, Z, R_y = 0)$. *recoverAll* is invoked in step 9. $\boxed{C}$

Execution of recoverAll

Steps 1 & 2 make no changes to Y and G
4. H is appended with $(P(R_z = 0|Y, Z, R_y = 0), G)$.
5. Flag is false and we proceed to the next step.
6. E is assigned FAIL since the query is not partially recoverable. 11. gmf function is invoked in line 11. It returns $(P(R_z = 0|Y, Z, R_y = 0), \emptyset, \emptyset)$. 12. *recFactors* is invoked and in line 1 an attempt to recover $P(R_z = 0|Y, Z, R_y = 0)$ is made again using *recoverAll*. However the query graph pair is contained in H and line 4 in *recoverAll* returns FAIL. Consequently, control now goes back to $\boxed{C}$ in function *recover_Dr*.

Execution of Function recover_Dr continuing from $\boxed{C}$

11. Failed is updated: $Failed = \{P(R_z = 0|Y, Z, R_y = 0)\}$. Since $Failed \neq \emptyset$, we proceed to step 13.
13. Function *getLatentVars* is invoked. $\{Y\}$ is the set of all partially observed variables in the query, $P(R_z = 0|Y, Z, R_y = 0)$, excluding Z. Since the factor corresponding to Y i.e. $P(R_y = 0|X, Y, Z)$ is not in Failed, t is not assigned 'false' in line 16. In line 17, since t is true and $Z \in S$, Latent=Z. Function returns Z.
14. (False, Z, false) is returned. Control now goes back to $\boxed{B}$ i.e. line 4 in $recFactors$.

Execution of Function recFactors continuing from $\boxed{B}$

8. Graph is changed to that shown in fig 2 (b).
9. *recoverAll* is invoked with the query $P(XY)$. In line 2 of *recoverAll*, the graph is pruned and the new graph is shown in fig 2 (c). H is appended with the new query graph pair and gmf is invoked again to yield: $(F_3 : P(X, Y, R_x = 0, R_y = 0), F_1 = \emptyset, F_2 = \{P(R_x = 0|X, Y, R_y = 0), P(R_y = 0|X, Y)\})$. As shown before these are recoverable and finally the estimand $\frac{P(X^*,Y^*,R_x=0,R_y=0)}{P(R_y=0)P(R_x=0|Y^*,R_y=0)}$ is returned. Control goes to $\boxed{A}$.

Execution of recoverAll continuing from $\boxed{A}$

12: $E \leftarrow \frac{P(X^*,Y^*,R_x=0,R_y=0)}{P(R_y=0)P(R_x=0|Y^*,R_y=0)}$
13. $E \neq FAIL$. Hence we move to the next step.
14. $\sum_{\mathbf{Z},\mathbf{Y}} \frac{\mathbf{P(X^*,Y^*,R_x=0,R_y=0)}}{\mathbf{P(R_y=0)P(R_x=0|Y^*,R_y=0)}}$ is returned.

The following example demonstrates a complex case of recoverability involving inducing paths.

Example 4 We will demonstrate recoverability given the following inputs: Query $P(X|D)$ m-graph given in figure 3 (a), $H = \emptyset$, $addvar = true$. In line 1, $pruneQ$ returns $\emptyset$ (since $X \perp\!\!\!\perp D$), thereby changing the query to $P(X)$. In line 2, graph is modified (by removing node D) as shown in figure 3 (b). Note that since D has been removed, R_D shall be treated as a fully observed variable. Since $addVar$ is true, $S = \{C, R_D, Y\}$. Obviously the query $P(X, Y, C, R_D)$ is not partially recoverable. In line 10, gmf returns $F_3 : P(X, Y, C, R_x = 0, R_y = 0, R_c = 0), F_1 = \emptyset, F_2 = \{P(R_x = 0|Y, R_D, R_y = 0, R_c = 0, C, X), P(R_y = 0|Y, R_D, R_c = 0, C, X), P(R_c = 0|Y, R_D, C, X)\}$. Function recFactors is invoked. Recoverability of all factors except $P(R_y = 0|Y, R_D, R_c = 0, C, X)$ is simple and identical to that discussed in example 3. We will hence discuss recoverability of $P(R_y = 0|Y, R_D, R_c = 0, C, X)$, starting from line 3 of *recover_Dr*.

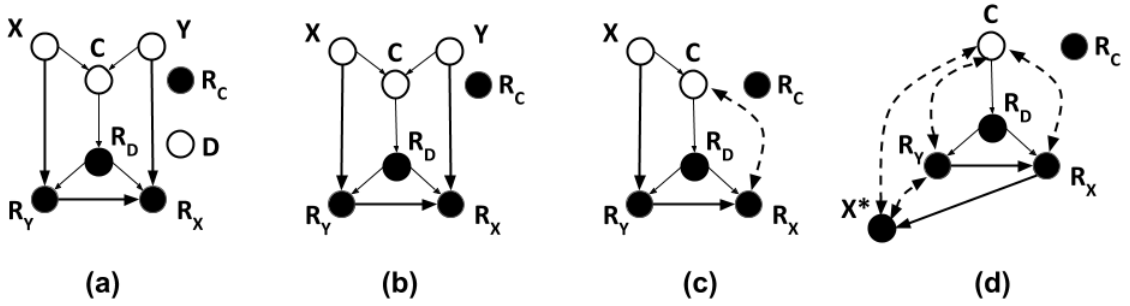

Figure 3: Intermediate graphs corresponding to Example 4

Execution of function *recover_Dr*

3. $P(R_y = 0|Y, R_D, R_c = 0, C, X)$ is pruned to $P(R_y = 0|R_D, X)$.
4. Since Y is no longer a part of the query now, the graph gets modified as shown in figure 3 (c).
5. No inducing paths are detected and we proceed to line 9.
9. Invoke recoverAll. In line 11 gmf further factorizes the query into $F_3 : P(R_y = 0|X, R_x = 0, R_D), F_1 = \{P(R_x = 0|X, R_D)\}, F_2 = \{P(R_x = 0|R_y = 0, X, R_D)\}$. In line 1 in recFactors, F_3 is recovered as $P(R_y = 0|X^*, R_x = 0, R_D)$. In line 4, recover_Dr is invoked. In line 5 of recover_Dr an inducing path is detected between X and R_x i.e. $IP = \{X\}$. Function *handleIP* transforms the parent query $P(R_y = 0|X, R_D)$ to $P(R_y = 0|X, \hat{R}_d)$, using rule-2 of do calculus. recoverAll is invoked in the last line of function *handleIP* $\boxed{D}$.

Execution of recoverAll

11. Function gmf returns: $F_3 = \{P(R_y = 0|X, R_x = 0, \hat{r_d})\}, F_2 = \{P(R_x = 0|X, R_y = 0, \hat{r_d})\}, F_1 = \{P(R_x = 0|X, \hat{r_d})\}$.
12. Function *recoverFactors* is invoked; in line 1 *recoverAll* is invoked to recover F_3. In line 6 of *recoverAll*, $E \leftarrow P(R_y = 0|X^*, R_x = 0, \hat{r_d})$. Since $E \neq FAIL$ and the query is causal, we proceed to line 9. Function *recoverCausal* is invoked $\boxed{E}$.

Execution of function recoverCausal

1. $Z^* = \{X^*\}$.
2. G^* is shown in figure 3 (d).
3. $E \leftarrow \frac{\sum_C P(X^*|R_x=0,R_y=0,C,r_d),P(R_x=0|R_y=0,C,r_d)P(R_y=0|C,r_d)P(C)}{\sum_{C,R_y} P(X^*|R_x=0,R_y,C,r_d),P(R_x=0|R_y,C,r_d)P(R_y|C,r_d)P(C)}$
5. Each factor in E is recoved using recoverAll.
9. Function returns $\frac{\sum_C P(X^*|R_x=0,R_y=0,C^*,R_c=0,r_d),P(R_x=0|R_y=0,C^*,R_c=0,r_d)P(R_y=0|C^*,R_c=0,r_d)P(C^*|R_c=0)}{\sum_{C,R_y} P(X^*|R_x=0,R_y,C^*,R_c=0,r_d),P(R_x=0|R_y,C^*,R_c=0,r_d)P(R_y|C^*,R_c=0,r_d)P(C^*|R_c=0)}$.
Control goes to $\boxed{E}$ in recoverAll.

In a similar manner factors $F_2 = \{P(R_x = 0|X, R_y = 0, \hat{r_d})\}$ and $F_1 = \{P(R_x = 0|X, \hat{r_d})\}$ are recovered. Thus the factor $P(R_y = 0|X, R_D)$ and hence $P(X)$ is recoverable.

Function recover_Dr($P(X|Y,\hat{d})$,S, F, G, H)

$Success \leftarrow \emptyset$, Failed $\leftarrow \emptyset$, $G^* \leftarrow G$
For every factor $f_{vi} \leftarrow P(R_{v_i} = 0|Z_i, \hat{d})$ in F
  $Z_i \leftarrow pruneQ(P(R_{v_i} = 0|Z_i, \hat{d}), G)$
  If $V_i \notin Z_i$ **then** $G^* \leftarrow$ getLatentGraph (V_i, G)
  $IP \leftarrow detectIP(P(R_{v_i} = 0|Z_i, \hat{d}), G^*)$
  If $IP \neq \emptyset$ **then**
    $E \leftarrow$ handleIP($P(X|Y,\hat{d}), IP, G^*, H$)
    If ($E \neq FAIL$) **then return** $(E, \emptyset, true)$
  $E \leftarrow$ recoverAll($P(R_{v_i} = 0|Z_i, \hat{d}), G^*, H, false$)
  If $E \neq$ FAIL **then** $Success \leftarrow Success \cup E$
  Else Failed $\leftarrow Failed \cup f_{vi}$
If Failed $== \emptyset$ **then return** $(Success, \emptyset, false)$
Latent $\leftarrow$ getLatentVars(S,Failed,G)
return (FAIL, Latent, false)

Function recover_Nr($P(X|Y,\hat{d}), S, F_1, F_2, G, H$)

$Success \leftarrow \emptyset$, Failed $\leftarrow \emptyset$, $G^* \leftarrow G$
For every factor $f_{vi} = P(R_{v_i} = 0|Z_i, \hat{d})$ in F_1
  $Z_i \leftarrow pruneQ(P(R_{v_i} = 0|Z_i, \hat{d}), G)$
  If $V_i \notin Z$ **then** $G^* \leftarrow$ getLatentGraph (V_i, G)
  Let $E_1 \in F_2$ be the recovered estimand of $P(R_{v_i} = 0|Z_i, X, \hat{d})$.
  If f_{vi} is recoverable using lemma 2, **then** let E denote the recovered estimand. go to step (8)
  $E \leftarrow$ recoverAll($P(R_{v_i} = 0|Z_i), G^*, H, false$)
  If $E \neq$ FAIL **then** $Success \leftarrow Success \cup E$
  else Failed $\leftarrow Failed \cup P(R_{v_i} = 0|Z_i, \hat{d})$
If Failed $== \emptyset$ **then return** $(Success, \emptyset)$
Latent $\leftarrow$ getLatentVars(S,Failed,G)
return (FAIL, Latent)

4. Conclusions

We exemplified recoverability in cases where a variable and its mechanism are connected by an inducing path. We presented a new factorization scheme that is general and applicable to both probabilistic and causal queries. Using this factorization scheme, we developed a general algorithm to recover conditional probability and interventional distributions.

Appendix A. Functions Invoked by Algorithm 1

Function pruneQ($\mathbf{P}(\mathbf{X}|\mathbf{Y},\hat{\mathbf{d}}), \mathbf{G}$)

If $Y == \emptyset$, **then return** Y
$\forall Y_1 \in Y$, **If** $X \perp\!\!\!\perp Y_1 | Y - \{Y_1\}, D$ in $G_{\overline{D}}$, **then** $Y \leftarrow Y - \{Y_1\}$
return Y

Function seen(H, Q, G)

If $(Q, G) \in H$ **then return** (H, TRUE)
$H \leftarrow H \cup (Q, G)$
return (H, FALSE)

Function gmf($\mathbf{P}(\mathbf{A}|\mathbf{B}, \hat{\mathbf{w}}), \mathbf{G}$)

$F_3 \leftarrow P(A|B, \hat{w}), F_2 \leftarrow \emptyset, F_1 \leftarrow \emptyset$
If $\nexists D_1 \in A_m \cap B_m$ such that $R_{D_1} \notin \{A, B, W\}$ **then return** $(P(A|B, \hat{w}), F_1, F_2)$
Let F_3 be $P(X|Y, R_D = 0, \hat{w})$ in eq 2 and F_1 and F_2 be lists containing factors on the RHS of equations 3 and 4 respectively.
Simplify F_1 and F_2 further by applying lemma 1.
return (F_3, F_1, F_2)

Function isParRec($P(X|Y, \hat{d}), G$)

If $X == \emptyset$, **return** $\emptyset$
$P(X|Y, \hat{d}) \leftarrow$ toProxy($P(X|Y, \hat{d})$)
$Z \leftarrow (X \cup Y) \cap V_m$
If $Z == \emptyset$ **then return** $P(X|Y)$
If $\{X, Y\} \cap R_Z \neq \emptyset$ **then return** FAIL
$R_Z \leftarrow R_Z - D$
If $X \not\perp\!\!\!\perp R_Z|Y, D$ in $G_{\overline{D}}$ **then return** FAIL
return $toProxy(P(X|Y \cup \{R_Z = 0\}, \hat{d}))$

The function isParRec checks if equation 1 is applicable to the input query, and if so, converts corresponding partially observed variables to proxy variables. For example, given $P(X_1, X_2|Y, R_y = 0, R_{X_1} = 0)$, it will return $P(X_1^*, X_2|Y^*, R_y = 0, R_{X_1} = 0)$ and given $P(X_1, X_2|Y, R_y = 0, R_{X_1} = 1)$, it will return $P(X_1, X_2|Y^*, R_y = 0, R_{X_1} = 1)$.

Function toProxy($\mathbf{P}(\mathbf{X}|\mathbf{Y}, \hat{\mathbf{d}})$)

$\forall Z \in Y \cap V_m$ such that $R_z \in Y$ and R_z assumes the value 0, $Y \leftarrow (Y - \{Z\}) \cup \{Z^*\}$
$\forall Z \in X \cap V_m$ such that $R_z \in Y \cup X$ and R_z assumes the value 0, $X \leftarrow (X - \{Z\}) \cup \{Z^*\}$
return $P(X|Y, \hat{d})$

Function getLatentVars($S, Failed, G$)

Latent $\leftarrow \emptyset$
For every factor $P(R_{v_i} = 0|Z_i, X, \hat{d})$ in Failed
    $t \leftarrow true$
    For every $V_j \in ((Z_i, X, D) \cap V_m) - V_i$
        If $P(R_{v_j} = 0|Z_j, \hat{d}) \in$ Failed **then** $t \leftarrow false$
    If t and $V_i \in S$ **then** $Latent \leftarrow Latent \cup V_i$
return Latent

Function addVars($\mathbf{P}(\mathbf{X}|\mathbf{Y}, \hat{\mathbf{d}}), \mathbf{G}$)

$S \leftarrow \emptyset$
If $D \neq \emptyset$ then $G \leftarrow getAncGraph(\{X, Y, D\}, G_{\overline{D}})$
$\forall Z \in V_m \cup V_o - \{X, Y, D\}$
    If Z is not a collider or descendant of a collider on any path between X_1 and R_{X_1} for any $X_1 \in X \cap V_m$, **then** $S \leftarrow S \cup \{Z\}$
    If Z is not a collider or descendant of a collider on any path between X_2 and R_{X_2} for any $X_2 \in V_m$ and $R_{X_2} \in X \cap R$, **then** $S \leftarrow S \cup \{Z\}$
return S

The following function constructs a latent projection (Verma and Pearl, 1991; Pearl, 2009; Shpitser et al., 2015) and returns the resulting graph G_l in which all variables in X will be treated as latent and not explicitly portrayed in the graph.

Function getLatentGraph($\mathbf{X}$, $\mathbf{G}$)

Do the following until no more edges can be added to G
  For all pairs of nodes Z, Y in G such that $\{Z, Y\} \cap X = \emptyset$
    If there exists a directed path from Z to Y such that all intermediate nodes belong to X and $Z \rightarrow Y$ is not an edge in G, **then** Add the edge $Z \rightarrow Y$
    If there exists a marginally d-connected path $Z \leftarrow ... \rightarrow Y$ such that all intermediate nodes belong to X and $Z \leftarrow\rightarrow Y$ is not an edge in G **then** Add the edge $Z \leftarrow\rightarrow Y$
  Remove all nodes in X from G
  $V_m \leftarrow V_m - X, U \leftarrow U \cup X$ $V^* \leftarrow V^* - X^*, R \leftarrow R - R_X,$ $V_o \leftarrow V_o \cup X^* \cup R_X$
  return G

Function getAncGraph($\mathbf{X}$, $\mathbf{G}$)

$\mathbf{Y} \leftarrow \mathbf{X}$
$\forall \mathbf{x} \in \mathbf{X}$, **if** $\exists \mathbf{R_x}$, **then** add $\mathbf{R_x}$ to $\mathbf{Y}$
Mark all $\mathbf{y} \in \mathbf{Y}$ in $\mathbf{G}$
$\mathbf{A} = \emptyset$
$\forall \mathbf{y} \in \mathbf{Y}$ add $parent(\mathbf{y})$ to $\mathbf{A}$, as long as $parent(\mathbf{y}) \notin \mathbf{Y}$
If $\mathbf{A} \neq \emptyset$ **then return** *getAncGraph*($\mathbf{A}$, $\mathbf{G}$)
Let G^* be the sub-graph of $\mathbf{G}$ comprising of all marked nodes in $\mathbf{G}$.
$\forall$ partially observed variables X in G^*, add to G^* proxy variable X^*, and the edges $R_x \rightarrow X^*$ and $X \rightarrow X^*$
$\forall R_x \in G^*$ such that $X \notin G^*$
  $R \leftarrow R - \{R_x\}$
  $V_o \leftarrow V_o \cup \{R_x\}$
return G^*

C-component of X (also known as district of X) is the set of variables (including X) that is connected to X by a path comprising bi-directed edges.

Function detectIP($P(X|Y, \hat{d}), G$)

$IP \leftarrow \emptyset$
$P(X|Y, \hat{d}) \leftarrow$ toProxy $(P(X|Y, \hat{d}))$
For every $X_1 \in X \cap V_m$
  If there exist inducing paths between X_1 and R_{X_1} in G **then** $IP \leftarrow IP \cup X_1$
For every $R_w \in X \cap R$
  If $W \in Y$ and there exist inducing paths between W and R_w in G **then** $IP \leftarrow IP \cup W$
return IP

Function handleIP($P(y|x, \hat{w}), Z, P, G, H$)

Let C denote the C-component of R_Z
$D \leftarrow$ $(Parents(C) - C) \cap Ancestors(R_Z)$
$\forall\ D_1 \in D$
  If $D_1 \in X$ & $P(y|x, \hat{w}) == P(y|x - d_2, \hat{d_2}, \hat{w})$ as per Rule 2 of do-calculus **then**
    $X \leftarrow X - \{D_1\}, W \leftarrow W \cup \{D_1\}$
  If $D_1 \notin X \cup Y$ & $P(y|x, \hat{w}) == P(y|x, \hat{d_1}, \hat{w})$ as per Rule 3 of do-calculus **then**
    $W \leftarrow W \cup \{D_1\}$
return $recoverAll(P(y|x, \hat{w}), G, H, false)$

References

R. Daniel, M. Kenward, S. Cousens, and B. De Stavola. Using causal diagrams to guide analysis in missing data problems. *Statistical Methods in Medical Research*, 21(3):243–256, 2012.

A. Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge University Press, 2009.

C. Enders. *Applied Missing Data Analysis*. Guilford Press, 2010.

R. Gill and J. Robins. Sequential models for coarsening and missingness. In *Proceedings of the First Seattle Symposium in Biostatistics*, pages 295–305. Springer, 1997.

J. Graham. *Missing Data: Analysis and Design (Statistics for Social and Behavioral Sciences)*. Springer, 2012.

D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. 2009.

R. Little and D. Rubin. *Statistical analysis with missing data*. Wiley, 2002.

K. Mohan and J. Pearl. Graphical models for recovering probabilistic and causal queries from missing data. In *Advances in Neural Information Processing Systems 27*, pages 1520–1528. Curran Associates, Inc., 2014.

K. Mohan, J. Pearl, and J. Tian. Graphical models for inference with missing data. In *Advances in Neural Information Processing Systems 26*, pages 1277–1285. 2013.

J. Pearl. *Causality: models, reasoning and inference*. Cambridge Univ Press, New York, 2009.

J. Robins. Non-response models for the analysis of non-monotone non-ignorable missing data. *Statistics in medicine*, 16(1):21–37, 1997.

D. Rubin. Inference and missing data. *Biometrika*, 63:581–592, 1976.

D. Rubin. Multiple imputation after 18+ years. *Journal of the American Statistical Association*, 91 (434):473–489, 1996.

I. Shpitser. Consistent estimation of functions of data missing non-monotonically and not at random. In *Advances in Neural Information Processing Systems*, pages 3144–3152, 2016.

I. Shpitser, K. Mohan, and J. Pearl. Missing data as a causal and probabilistic problem. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, 2015.

T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Conference in Artificial Intelligence*, pages 220–227. Association for Uncertainty in AI, 1991.