2.5 Recovering DAG Structures

With the added assumption of stability, every distribution has a unique minimal causal structure (up to d-separation equivalence), as long as there are no hidden variables. This uniqueness follows from Theorem 1.2.8, which states that two causal structures are equivalent (i.e. they can mimic each other) if and only if they relay the same dependency information—namely, they have the same skeleton and same set of v-structures.

In the absence of unmeasured variables, the search for the minimal model then boils down to reconstructing the structure of a DAG D from queries about conditional independencies, assuming that those independencies reflect d-separation conditions in some undisclosed underlying DAG D_0. Naturally, since D_0 may have equivalent structures, the reconstructed DAG will not be unique, and the best we can do is to find a graphical representation for the equivalence class of D_0. Such graphical representation was introduced in Verma and Pearl (1990) under the name *pattern*. A pattern is a partially directed DAG, in particular, a graph in which some edges are directed and some are nondirected. The directed edges represent arrows that are common to every member in the equivalence class of D_0, while the undirected edges represent ambivalence; they are directed one way in some equivalent structures and another way in others.

The following algorithm, introduced in Verma dna Pearl (1990), takes as input a stable probability distribution $\hat{P}$ generated by some underlying DAG D_0 and outputs a pattern that represents the equivalence class of D_0.[7]

IC Algorithm (Inductive Causation)

Input: $\hat{P}$, a stable distribution on a set V of variables.

Output: a pattern $H(\hat{P})$ compatible with $\hat{P}$.

1. For each pair of variables a and b in V, search for a set S_{ab} such

[7] The IC algorithm, as introduced in Verma and Pearl (1990), was designed to operate on latent structures. For clarity, we here present the algorithm in two separate parts, IC and IC*, with IC restricted to DAGs and IC* operating on latent structures.

that $(a \perp\!\!\!\perp b | S_{ab})$ holds in $\hat{P}$—in other words, a and b should be independent in $\hat{P}$, conditioned on S_{ab}. Construct an undirected graph G such that vertices a and b are connected with an edge if and only if no set S_{ab} can be found.

2. For each pair of nonadjacent variables a and b with a common neighbor c, check if $c \in S_{ab}$.
   If it is, then continue.
   If it is not, then add arrowheads pointing at c (i.e., $a \rightarrow c \leftarrow b$).

3. In the partially directed graph that results, orient as many of the undirected edges as possible subject to two conditions: (i) the orientation should not create a new v-structure; and (ii) the orientation should not create a directed cycle.

The IC algorithm leaves the details of steps 1 and 3 unspecified, and several refinements have been proposed for optimizing these two steps. Verma and Pearl (1990) noted that, in sparse graphs, the search can be trimmed substantially if commenced with the Markov network of $\hat{P}$, namely, the undirected graph formed by linking only pairs that are dependent conditionally on all other variables. In linear Gaussian models, the Markov network can be found in polynomial time, through matrix inversion, by assigning edges to pairs that correspond to the nonzero entries of the inverse covariance matrix. Spirtes and Glymour (1991) proposed a general systematic way of searching for the sets S_{ab} in step 1. Starting with sets S_{ab} of cardinality 0, then cardinality 1, and so on, edges are recursively removed from a complete graph as soon as separation is found. This refinement, called the PC algorithm (after its sauthors, Peter and Clark), enjoys polynomial time in graphs of finite degree because, at every stage, the search for a separating set S_{ab} can be limited to nodes that are adjacent to a and b.

Step 3 of the IC algorithm can be systematized in several ways. Verma and Pearl (1992) showed that, starting with any pattern, the following four rules are required for obtaining a maximally oriented pattern.

R_1: Orient b–c into $b \rightarrow c$ whenever there is an arrow $a \rightarrow b$ such that a and c are non adjacent.

R_2: Orient a–b into $a \rightarrow b$ whenever there is chain $a \rightarrow c \rightarrow b$.

R_3: Orient a–b into $a \rightarrow b$ whenever there are two chains a–$c \rightarrow b$ and a—$d \rightarrow b$ such that c and d are nonadjacent.

R_4: Orient a–b into $a \rightarrow b$ whenever there are two chains a–$c \rightarrow d$ and $c \rightarrow d \rightarrow b$ such that c and b are nonadjacent.

Meek (1995) showed that these four rules are also sufficient, so that repeated application will eventually orient *all* arrows that are common to the equivalence class of D_0. Moreover, R_4 is not required if the starting orientation is limited to v-structures.

Another systematization is offered by an algorithm due to Dor and Tarsi (1992) that tests (in polynomial time) if a given partially oriented acyclic graph can be fully oriented without creating a new v-structure or a directed cycle. The test is based on recursively removing any vertex v that has the following two properties:

1. no edge is directed outward from v;

2. every neighbor of v that is connected to v through an undirected edge is also adjacent to all the other neighbors of v.

A partially oriented acyclic graph has an admissible extension in a DAG if and only if all its vertices can be removed in this fashion. Thus, to find the maximally oriented pattern, we can (i) separately try the two orientations, $a \rightarrow b$ and $a \leftarrow b$, for every undirected edge a—b, and (ii) test whether both orientations, or just one, have extensions. The set of uniquely orientable arrows constitutes the desired maximally oriented pattern. Additional refinements can be found in Chickering (1995), Andersson et al. (1997), and Moole (1997).

Latent structures, however, require special treatment, because the constraints that a latent structure imposes upon the distribution cannot be completely characterized by any set of conditional independence statements. Fortunately, certain sets of those independence constraints can be identified [Verma and Pearl, 1990]; permits us to recover valid fragments of latent structures.