Assignment 3

Hand in by May 12 to Søren (soren.wengel_mogensen@control.lth.se).

You need to solve Problem A or Problem B. You do not need to solve both. Hints are provided at the end of this document.

Problem A In this problem, we will prove that the latent projection is in fact a 'marginal'. Let $\mathcal{D} = (V, E)$ be a DAG and assume that O and H are disjoint set such that $V = O \cup H$. Let $m(\mathcal{D}, O)$ denote the latent projection of $\mathcal{D}$ on the set O as defined in the slides from Week 6.

Let $\mathcal{G} = (V, E)$ be an ADMG and let $i, j \in V, i \neq j$, $C \subseteq V \setminus \{i, j\}$. We say that a path between i and j in $\mathcal{G}$ is *m-connecting* given C if every collider on the path is an ancestor of C (recall that $C \subseteq an(C)$) and no noncollider on the path is in C. If there are no m-connecting paths between i and j given C in $\mathcal{G}$, then we say that i and j are *m-separated* given C in $\mathcal{G}$. We are now ready to state the task.

Prove that for all $i, j \in O$, $i \neq j$, and $C \subseteq O \setminus \{i, j\}$ it holds that i and j are d-separated by C in $\mathcal{D}$ if and only if i and j are m-separated by C in $m(\mathcal{D}, O)$. If you wish, you can extend your proof to the case where $\mathcal{D}$ is a general ADMG, and not necessarily a DAG.

Problem B In this problem, we will implement an algorithm to compute the latent projection of a DAG, $\mathcal{D} = (V, E)$. Your code should take $\mathcal{D}$ and a set $O \subseteq V$ as input and output $m(\mathcal{D}, O)$ as defined in the slides from Week 6. Below we describe one algorithm to do this, but you may of course also do this in other ways. In the figure, there are four DAGs. Please hand in the latent projection of these four graphs along with your implementation. We now give a high-level description of an algorithm you can use to compute a latent projection. Note that you do not need to prove that this algorithm outputs the latent projection – you only need to implement it.

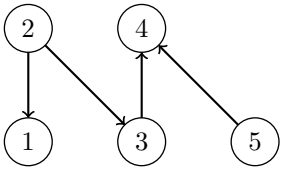
In an ADMG, $\mathcal{G} = (V, E)$, we say that an ordered triple of nodes (k, l, m), $k, l, m \in V$, is *unshielded* if $l \in V \setminus O$ and (1), (2), or (3) holds.

(1) $k \to l \to m$ is in $\mathcal{G}$ and $k \to m$ is not in $\mathcal{G}$.

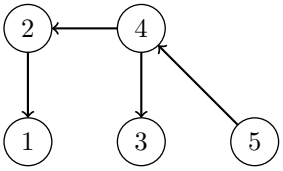
(2) $k \leftarrow l \to m$ is in $\mathcal{G}$ and $k \leftrightarrow m$ is not in $\mathcal{G}$.

(3) $k \leftarrow l \leftrightarrow m$ is in $\mathcal{G}$ and $k \leftrightarrow m$ is not in $\mathcal{G}$.

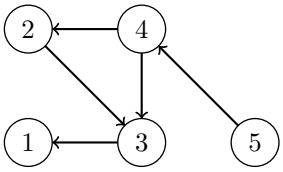
Note that a triple, (k, l, m), is only unshielded if $l \in V \setminus O$. However, k and m could be in O or in $V \setminus O$.



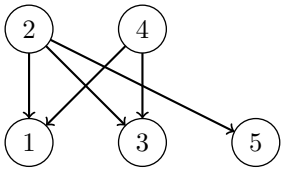
(a) DAG 1. Use $O = \{1, 3, 5\}$.



(b) DAG 2. Use $O = \{1, 3, 5\}$.



(c) DAG 3. Use $O = \{1, 3, 5\}$.



(d) DAG 4. Use $O = \{1, 3, 5\}$.

Figure 1: Example DAGs for which to compute their latent projections on the set $O = \{1, 3, 5\}$.

We now define an *edge addition*. If (k, l, m) is an unshielded triple and (1) holds, then an edge addition adds $k \to m$ to the graph. If (k, l, m) is an unshielded triple and (2) or (3) hold, then an edge addition adds $k \leftrightarrow m$ to the graph.

The algorithm works as follows. Starting from $\mathcal{D}$, we find an unshielded triple and perform an edge addition to obtain a new graph $\mathcal{D}_1$ with one more edge. We repeat this to obtain a sequence of graphs $\mathcal{D}, \mathcal{D}_1, \ldots, \mathcal{D}_N$ and finish when there are no more unshielded triples in the current graph. We then define $\mathcal{G} = (O, F)$ to be the ADMG such that for $i, j \in O$, the edge $i \to j$ is in $\mathcal{G}$ if and only if $i \to j$ is in $\mathcal{D}_N$ and such that $i \leftrightarrow j$ is in $\mathcal{G}$ if and only if $i \leftrightarrow j$ is in $\mathcal{D}_N$. The graph $\mathcal{G}$ is the latent projection of $\mathcal{D}$ on O, $m(\mathcal{D}, O)$.

Hints

Problem A: Consider first a d-connecting path between i and j given C in $\mathcal{D}$. Argue that you can construct an m-connecting path in the latent projection by 'collapsing' the parts of the path that are outside O (these subpaths will generate edges in the latent projection).

Consider then an m-connecting path between i and j given C in $\mathcal{G}$. Argue that you can 'expand' the edges into paths in $\mathcal{D}$ and compose these to obtain a d-connecting path in $\mathcal{D}$.

Problem B: You can represent an ADMG using two adjacency matrices, one for directed edges and one for bidirected edges. The one representing the bidirected edges should be a symmetric matrix. A DAG is, of course, also an ADMG, but has all zeros in the bidirected adjacency matrix.