ECAI 2020
G.D. Giacomo et al. (Eds.)

doi:10.3233/FAIA200396

Local to Global Learning of a Latent Dynamic Bayesian Network

Dan Halbersberg and **Boaz Lerner** [1]

Abstract. Latent variables (LVs) represent the origin of many scientific, social, and medical phenomena. While models with only observed variables (OVs) have been well studied, learning a latent variable model (LVM) allowing both types of variables is difficult. Therefore, the assumption of no LVs is usually made, but modeling by ignoring LVs leads to learning a partial/wrong and misleading model that misses the true realm. In recent years, progress has been made in learning LVMs from data, but most algorithms have strong assumptions limiting their scope. Moreover, LVs by nature often change temporally, adding to the challenge and complexity of learning, but current LVM learning algorithms do not account for this. We propose learning locally a causal model in each time slot, and then local to global learning over time slices based on probabilistic scoring and temporal reasoning to transfer the local graphs into a latent dynamic Bayesian network with intra- and inter-slice edges showing causal interrelationships among LVs and between LVs and OVs. Examined using data generated synthetically and of ALS and Alzheimer patients, our algorithm demonstrates high accuracy regarding structure learning, classification, and imputation, and less complexity.

1 Introduction

Our complex world comprises observed and latent variables [17]. While the term latent variables (LVs) is mainly used in neural networks, clustering, and factor analysis for dimensionality reduction, where there is no semantic or physical meaning to the result, we focus on latent causal structure learning, where LVs are medically, psychologically, or sociologically important, e.g., stress, mood, and satisfaction. LVs and their interrelations can explain root causes for values measured by observed variables (OVs). We consider observed variables as proxies of LVs, and employ them to discover causal pathways connecting LVs to themselves and to OVs. The multiple indicator model (MIM) [4] presents such pathways, but learning an MIM usually requires knowing the correct causal relations between LVs and OVs (i.e., the measurement model) [19], and if this is missing, algorithms, e.g., MIMBuilt [27], which learn the causal relations between LVs (i.e., the structural model) based on a given (assumed correct) measurement model, may be misleading. We are interested in learning a latent variable model (LVM) from data without knowing in advance the number of LVs, nor their cardinality (number of states), interrelations, or relations to the OVs that measure them. Finding causal relations manifested in an LVM is difficult because the joint distribution can be generated by an infinite number of different LVMs.

Several algorithms have been proposed to learn a static LVM, such as structural expectation maximization (SEM) [10], FindHidden [8], fast causal inference (FCI) [29], a family of algorithms to learn LV trees [13, 35], BuildPureClusters (BPC) [27], find one factor clusters [18], find two factor clusters [19], and learning pairwise cluster comparison (LPCC) [2]. These algorithms usually need information about the number of LVs and their cardinality [2], and only find evidence for the existence of LVs, but do not necessarily learn an LVM [27], or provide only a partial causal explanation due to limiting assumptions.

Learning a dynamic (temporal) LVM, where LVs and their interrelations may change in time, is more challenging. A common dynamic graphical model is the hidden Markov model (HMM) [25] and its extensions, the factorial HMM [12] and dynamic Bayesian network (DBN) [24], that generalize it under the stationarity assumption, representing multi-variable models. However, DBN learning is NP-hard [24, 30], requiring a greedy search. Also, learning with data missing, which is the essence behind learning an LVM, requires an EM or a gradient method, which are computationally expensive. Last, like for the (static) LVM learned using search and score algorithms (e.g., SEM, FindHidden), the learned DBN may fit the data well, but miss causal relations. tsFCI [9] learns a dynamic model from time-series data using a causal structure learning method, but it neither explicitly identifies LVs, nor their values, nor the relationships among them.

In this paper, we replace the common expensive (EM) greedy search for a locally optimized DBN with inexpensive (a single EM call) global temporal learning based on local causal MIM structures learned by LPCC. Our contribution is: 1) an algorithm that learns local static causal structures for each time slice of a sequence, each of which is asymptotically a pattern of the true static graph, and combines the local structures into a dynamic model by probabilistic scoring and temporal-causal reasoning, forming a stationary temporal MIM-based LVM; 2) performance measures of structural correctness for a static LVM and their extension to a temporal LVM; and 3) in showing theoretical correctness of our algorithm and providing empirical evidence for its usefulness and accuracy in: a) learning synthetic temporal graphs, b) inducing dynamic latent-state prediction models, c) performing data imputation in missingness scenarios, and d) representing latent medical knowledge for better understanding of neurodegenerative diseases. Following, we briefly describe the LPCC (Sec. 2), which is our local structure learning algorithm, how we turn local LVM structure learning into temporal LVM learning (Sec. 3), and our empirical evaluation (Sec. 4), before summarizing (Sec. 5).

2 Learning pairwise cluster comparison

First are two definitions related to LVM. Capital/small letters denote variable/value, and bold is used for vectors.

Definition. ***Measurement model***—A directed acyclic graph (DAG) over sets of OVs **O**, LVs **L**, and edges is a measurement model if a variable in **L** is a parent of at least one variable in **O**, a variable in **O**

[1] Ben Gurion University of the Negev, Israel, emails: halbersb@bgu.ac.il, boaz@bgu.ac.il

is a child of at least one variable in **L**, and none of the variables in **O** is a parent of any variable in **L** [27].

Definition. ***Pure measurement model***—A measurement model is called a pure measurement model (PMM) iff each variable in **O** has a single parent and that parent is in **L**.

Second, we present the learning pairwise cluster comparison (LPCC) algorithm used to locally learn a PMM for each time slice (Sec. 3.1). The LPCC algorithm [2] links learning a causal graphical model and cluster analysis such that the structure of the model is learned based on the analysis of the clusters. The main idea is that a change in the value of an exogenous LV is reflected in the values taken by its OV descendants, and thus, if we want to identify an exogenous LV, we should check for value changes in its OVs. To do that, we first define: 1) *exogenous variable* as zero in-degree (i.e., a variable that has no parents in the graph) 2) a *major effect* as that maximizing the conditional probability of an OV value given a specific configuration of all its exogenous ancestors, 3) a *major value* as the value OV takes when the effect is major, and 4) a *major cluster* as the data cluster that corresponds to the OV's major value, and thus, represents the major effect of the specific exogenous configuration on this OV. Note that the set of all major clusters (each corresponds to a major value of an OV due to its exogenous configuration) reflects the effect of all possible exogenous value configurations, and thus the number of major clusters equals the number of all configurations of all exogenous variables. Thus, major cluster identification is fundamental to the discovery of exogenous variables and (later) their causal interrelations.

The LPCC algorithm performs pairwise cluster comparison (PCC) between any two major clusters; these are represented by their centroids of dimension $|\mathbf{O}|$ (the number of OVs). The result of a PCC is a binary vector of size $|\mathbf{O}|$, in which each element is 1 or 0 depending, respectively, on whether or not there is a difference between the corresponding elements in the compared major cluster centroids. Note that it is only the PCC result that is binary, not the data or the centroids. Thus, a PCC element of 1 identifies an OV that has changed its value between the compared clusters due to a change in its ancestor LVs, which is an evidence of causal relationships between the two (see proof in [2]). Therefore, each PCC reveals a value configuration of the exogenous LVs and, together, all PCCs reveal the LVs themselves. Similarly, the relationships among the LVs can be learned.

To demonstrate a PCC, Table 1 shows centroids of four major clusters and the corresponding PCCs. The centroids are of clusters found for six-dimensional records sampled from the graph in Fig. 1(a) having two binary LVs and six binary OVs (binary for the example, as LPCC is not restricted by variable cardinality). To reveal each exogenous LV, we have to check which elements in a PCC change *always together*, as this reflects a real change in the exogenous LV.

Definition. ***A maximal set of observed*** (MSO) variables is the maximal set of OVs that always change values together in all PCCs.

Identification of MSOs is based on the PCC matrix. Table 1b shows that $X1, X2$, and $X3$ always change values together, as do $X4, X5$, and $X6$, forming two MSOs that each include descendants of the same LV, introducing, in this example, two exogenous LVs, as shown in Fig. 1(a) from which the data were sampled. Since every OV belongs to a single MSO, i.e., MSOs corresponding to different LVs are disjoint, LPCC learns a PMM [2].

The LPCC algorithm learns an LVM in two stages. In the first, it identifies exogenous latents and latent colliders, along with their descendants. In a collider (V-structure) [e.g., L_2 in Fig 1(b)], two LVs have a mutual child LV, and they are conditionally dependent given this child [14]. An LV L_i is a latent collider iff 1) its indicators (MSO) change value together sometimes with latent L_j indicators and sometimes with L_k indicators, and 2) its MSO never changes value alone in any PCC (i.e., its indicators are never the only "1"s in any PCC) [2]. The intuition is that a latent collider would change value iff at least one of its parents changed value.

Table 2a shows four major cluster centroids of six-dimensional clustered data sampled from the six binary (binary only for the example) OVs of the graph in Fig. 1(b). Table 2b demonstrates the corresponding PCCs in which $X1$ and $X2$, $X3$ and $X4$, and $X5$ and $X6$ always change values together as part of the first, second, and third MSOs, respectively. Note that the second MSO changes value sometimes only with the first MSO (row 4), sometimes only with the third MSO (row 2), and sometimes with both (row 6), but never alone. Thus, the LPCC learns that L_2 (with OVs $X3$ and $X4$) is a latent collider, and the causal graph is Fig. 1(b), from which indeed the data was generated.

In the LPCC second stage, non-collider LVs are split from previously learned exogenous latent ones (for more details see [2]).

3 Combining local graphs

Local to global learning (LGL) of a Bayesian network (BN) was introduced in [21] and later expanded, e.g., in [1], suggesting to learn locally (separately) the Markov blanket of each variable in the BN and to combine all local graphs into the global graph.

LGL can be implemented using simple edge scoring, $E_{i,j} = \sum_m E^m_{i,j}/|M|$, where $E^m_{i,j} \in \{0,1\}$ is the edge between nodes i and j in the $m \in M$ learned graph (all M graphs can be learned in parallel), contributing 1 if it exists in graph m and 0 otherwise. Once all edges are scored and ranked, an algorithm can add edges to the initial graph starting from a high to a low score, as long as the added edge does not violate the directed acyclic graph property and the score increases.

However, LGL may work only with a BN and fully observed data, but in the presence of LVs, it will lead to a wrong Markov blanket (an LV d-connects OVs for which it is a common cause [29]). Also, edge scoring is not applicable since the number of LVs and their identities may change across local graphs. Graphs (a) and (b) in Fig. 1 (say locally learned for two time slices) have different numbers of LVs and represent different concepts for each LV, and thus, e.g., edge $L_2 \rightarrow X_4$ in both graphs refers to different concepts, and combining scores for this edge across the two graphs has no meaning.

Next, we introduce a new algorithm to dynamically learn local graphs in the presence of LVs and to merge them into a global LVM. A local MIM is learned by the LPCC sequentially for each time slice, and then local MIM-based LVMs are combined by probabilistic scoring and temporal-causal reasoning to make the global latent DBN.

3.1 The LGL-based LPCC algorithm

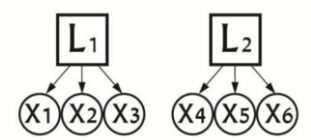
Our proposed LGL algorithm assumes three assumptions:

Assumption 1. ***Stationary model***—Data (and the process/model that has generated them) are stationary and discrete.

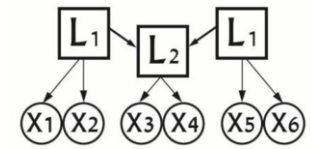
Learning time-varying models usually assumes no LVs [22, 28, 26] because an LV d-connects OVs of subsequent slices (these OVs may be the same OV in the two slices), undermining parameter learning and inference. Thus, to learn a temporal LVM, we require stationarity.

Assumption 2. ***DBN model***—The model is a DBN, i.e., a pair $(BN_1, BN_{\rightarrow})$, where BN_1 defines the BN in $t = 1$ and $BN_{\rightarrow}$ is a two-slice temporal BN (2-TBN) with LVs.

Assumption 3. ***PMM***—Each slice in the 2-TBN is a PMM (this is needed *only* if local graphs are learned by LPCC).



(a) LVM with two exogenous LVs



(b) LVM with two exogenous LVs and a latent collider

Figure 1: (a) and (b) LVMs that are learned using the PCCs in Tables 1 and 2, respectively.

Table 1: a) Centroids of four major clusters for data sampled from Fig. 1(a). b) The corresponding PCCs.

Centroid	X1	X2	X3	X4	X5	X6
C1	0	0	0	1	1	1
C2	1	1	1	1	1	1
C3	0	0	0	0	0	0
C4	1	1	1	0	0	0

(a)

PCC	δX1	δX2	δX3	δX4	δX5	δX6
1-2	1	1	1	0	0	0
1-3	0	0	0	1	1	1
1-4	1	1	1	1	1	1
2-3	1	1	1	1	1	1
2-4	0	0	0	1	1	1
3-4	1	1	1	0	0	0

(b)

Table 2: a) Centroids of four major clusters for data sampled from Fig. 1(b). b) The corresponding PCCs.

Centroid	X1	X2	X3	X4	X5	X6
C1	0	0	1	1	0	0
C2	1	1	1	1	1	1
C3	0	0	0	0	1	1
C4	1	1	1	1	0	0

(a)

PCC	δX1	δX2	δX3	δX4	δX5	δX6
1-2	1	1	0	0	1	1
1-3	0	0	1	1	1	1
1-4	1	1	0	0	0	0
2-3	1	1	1	1	0	0
2-4	0	0	0	0	1	1
3-4	1	1	1	1	1	1

(b)

The PMM assumption is justified in many real-world applications in which concepts have their own indicators, such as: 1) a topic model, where each key phrase (e.g., soccer) is an indicator of a single topic (e.g., sport) [23]; 2) audio-visual speech recognition, where "mouth" and "silence" detectors and "skin" and "texture" detectors are the measures of "audio" and "visual" LVs, respectively, and both are children of a third latent, "speaker" [11]; 3) psychology, especially for data obtained by questionnaires designed to target specific latent factors (e.g., stress, satisfaction) [27]; 4) diagnosis of diseases, each based on its own symptoms; 5) marketing [36]; and 6) explaining the involvement of young drivers in road accidents [3]. A practical motivation to learning PMMs is that their equivalence class is much smaller than that of MIMs, which are indistinguishable [27]. When the PMM assumption is violated, our algorithm will learn a pure sub-model of the true model.

We propose LGL of local 2-TBN models, one learned for each time slice, to compose a global stationary LVM. Each 2-TBN is of two consecutive time slices to allow learning both inter (between-slice) and intra (within-slice) edges. The parameters of each local graph are estimated using the backward/forward algorithm (parameters must be tied across time slices since LVs d-connect the entire observed sequence [24]). Since local learning is based on the LPCC algorithm, we call our algorithm an *LGL-based LPCC*.

To accommodate the temporal setting and be able to evaluate learning a temporal LVM, we score an edge by the support it gets by all local 2-TBNs, $E_{i,j} = \sum_t E^t_{i,j}/(T-1)$, where $E^t_{i,j}$ in an indicator variable for the existence of an edge between nodes i and j (the former represents an LV and the second either an LV or an OV) in the local graph learned in time slice t of T slices. This score probabilistically accounts for co-occurrence of variables connected by an edge across local graphs, implying co-existence of these variables. In its first and second stages, the LGL-based LPCC algorithm globally learns the most probable connections (over all local graphs) between latent parents and their observed children (i.e., the measurement model) and among the LVs (i.e., the structural model), respectively.

In the first stage, we introduce a matrix S, holding the number of times any two OVs, O_i and O_j, share the same latent parent (are in the same MSO) in all local graphs,

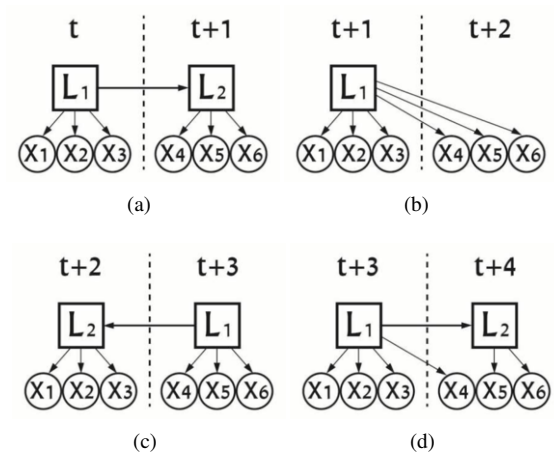
$$S(i,j) = \sum_{t=1}^{T-1} I(L(O_i^{G_t}) == L(O_j^{G_t})),$$

where $I(\cdot)$ is the indicator function, and $L(O_i^{G_t})$ is the latent parent of O_i in G_t (under the PMM assumption, an OV has only a single parent that is latent). The most probable members in O_i's MSO over all local graphs

$$LP_i = find\{S(i,:) == \arg\max_j S(i,:) : \forall j\},$$

create O_i's list of *local partners* (LPs), composing LP_i. Those OVs co-existing in each other's LP list make up the list of *global partners*, GPs, with its corresponding latent parent. Thereby, the first stage transforms MSOs of local graphs, each MSO with its corresponding local latent parent, to MSOs of the global graph, each such MSO with its corresponding global latent parent.

In the second stage of the LGL-based LPCC, we introduce a co-occurrence matrix, CO, that holds the probabilities for edges between any two LVs in the local graphs. Since any such edge can be oriented in either direction in different local graphs, its most probable direction over the local graphs is learned. Estimation of all CO's probabilities yields a 2-TBN reflecting all global LV—LV connections.



(a) (b) (c) (d)

Figure 2: Local graphs with one LV and 3 OVs over 5 slices.

Fig. 2 demonstrates learning an example temporal model with a single LV and three OVs using four local graphs over five slices $(t \cup t+1)$, $(t+1 \cup t+2)$, $(t+2 \cup t+3)$, and $(t+3 \cup t+4)$. The algorithm converts these local graphs into a global 2-TBN in two stages (*numbers* in brackets are lines in Algorithm 1):

1. In the first stage, matrix S counts the number of times each two OVs share a latent parent across all local graphs (*2–7*). Table 3a

Input: *(T-1) local graphs learned by LPCC*, **O**
Output: *2-TBN*
$S = zeros(|\mathbf{O}|,|\mathbf{O}|)$;
foreach *local graph* **do**
  foreach *LV* **do**
    foreach *two OVs in LV's MSO* **do**
      i = *index of the first OV*;
      j = *index of the second OV*;
      $S(i,j) = +1$;
LP = *empty list*(|**O**|,1);
foreach OV_i *in* **O** **do**
  $LP(i)$ = *empty list*;
  $LP(i) = find(S(i,:) == max[S(i,:)])$;
GP = *empty list of global MSOs*;
foreach OV_i *in* **O** **do**
  if $i \in GP$ **then**
    j = *index in GP where i exists*;
    foreach *variable k in* $LP(i)$ **do**
      if $k \notin GP(j)$ **then**
        Add k to $GP(j)$;
  else
    $GP(end) = [i \quad LP(i)]$;
2-TBN = *zeros*(|**O**| + |**GP**|,|**O**| + |**GP**|);
Add an edge in the 2-TBN between each latent and its global MSO of observed children in GP;
$CO = zeros(|\mathbf{GP}|,|\mathbf{GP}|)$;
foreach *local graph* **do**
  foreach *edge between head and tail LVs* **do**
    i = *index of the global MSO in GP that its intersection with the local MSO of the tail latent is the largest*;
    j = *index of the global MSO in GP that its intersection with the local MSO of the head latent is the largest*;
    $CO(i,j) = +1$;
Rank all edges by their counts in CO;
foreach *ranked edge in CO from high to low* **do**
  if *edge does not violate the DAG property* **then**
    Add the edge to 2-TBN;

Algorithm 1: Conversion of local LVMs to a global 2-TBN.

shows that $X3$ and $X4$ in Fig. 2 share an LV in two graphs [Figs. 2(b) and 2(d)], and thus, $S(3,4) = S(4,3) = 2$.

2. List LP_i for OV_i is created (*10*) to hold those most probable OVs that share with OV_i the same latent parent (*11*). Lists due to Table 3a are:
   1) $LP1$: $X1 \Rightarrow X2, X3$ 4) $LP4$: $X4 \Rightarrow X5, X6$
   2) $LP2$: $X2 \Rightarrow X1, X3$ 5) $LP5$: $X5 \Rightarrow X6$
   3) $LP3$: $X3 \Rightarrow X1, X2$ 6) $LP6$: $X6 \Rightarrow X5$.
3. After the global MSO list, GP, is initialized (*12*), it is filled (*13–20*) with LVs and their corresponding global MSOs, which are taken from the corresponding local MSO lists' LPs. In our example, $X1$, $X2$, and $X3$ in $LP1$ are added first in GP. The next two lists ($LP2, LP3$) have no effect on GP. $X5$ and $X6$ are fed next ($LP5$), before being merged with $X4$ ($LP4$) (due to transitivity). Finally, GP is: $\{\{X1, X2, X3\}, \{X4, X5, X6\}\}$, with each of the two global MSOs connected to its latent parent to form the (measurement model of the) 2-TBN (*22*).
4. In the second stage (*23–32*), the co-occurrence matrix between LVs, CO, is determined based on the local graphs. According to the first locally learned graph (Fig. 2a), $L1 \rightarrow L2$; thus, the $(1,2)$ entry in Table 3b increases by 1 (*28*). Note that according to GP, $L1$ and $L2$ are associated with $\{X1, X2, X3\}$ and $\{X4, X5, X6\}$, respectively. The second local graph (Fig. 2b) adds no information about the relation between $L1$ and $L2$. The third local graph (Fig. 2c) also shows $L2 \leftarrow L1$, but since the connections of the OVs to the LVs are opposite to those in GP, there is no support for $L1 \rightarrow L2$, but for $L2 \rightarrow L1$, and thus, entry $(2,1)$ in Table 3b increases by 1. The fourth local graph (Fig. 2d) increases the $(1,2)$ entry in Table 3b by 1 (*28*) because three of the four variables in $L1$'s local MSO, $\{X1, X2, X3, X4\}$, are part of $L1$'s global MSO in GP, $\{X1, X2, X3\}$, providing stronger evidence than when only one variable, $\{X4\}$, in $L1$'s local MSO, $\{X1, X2, X3, X4\}$, is part of $L2$'s global MSO in GP, $\{X4, X5, X6\}$. Because entry $(1,2)$ is larger than entry $(2,1)$, the algorithm (*29*) learns $L1 \rightarrow L2$ in the final graph (in case of a tie, the edge remains undirected). Note that if the first and third local graphs were examined in the opposite order, we would have obtained the symmetric matrix to Table 3b, but learned the same resulting graph ($''L1''$ and $''L2''$ are arbitrary, indicated by LPCC).

Table 3: Regarding Fig. 2: a) Matrix S counts local graphs in which a pair of OVs shares the same LV parent. b) Matrix CO holds the co-occurrence counts of edges between LVs.

	X1	X2	X3	X4	X5	X6
X1	-	4	4	2	1	1
X2	4	-	4	2	1	1
X3	4	4	-	2	1	1
X4	2	2	2	-	3	3
X5	1	1	1	3	-	4
X6	1	1	1	3	4	-

(a)

	L1	L2
L1	-	2
L2	1	-

(b)

To ensure causal reasoning by the global temporal LVM, a preprocessing stage precedes the algorithm in which two orientation rules are applied to local graphs. First, a learned inter-slice edge is oriented from t to $t+1$ because an event can only causally affect a future event [20]. Second, if a latent collider is discovered in a local graph in time $t+1$, then undirected edges in this graph in time t (recall that the local graph is a pattern of the true graph) are oriented along the directed edges of that collider due to its strong causal evidence [29]. Nevertheless, based on the co-occurrence matrix, CO, the LGL-based algorithm may learn a partial directed acyclic graph because if matrix entry (i,j) between two LVs i and j equals entry (j,i), then the edge is left undirected. Lemma 1 proves that our algorithm learns a PMM.

Lemma 1. *The global 2-TBN learned by the LGL-based LPCC algorithm is a PMM.*

Proof. By elimination. Assume the global 2-TBN is not pure, i.e., there is an O_i that has more than one parent (say two). Since the LGL algorithm only learns edges that are directed from LVs, these two parents must be latent, say L_j and L_k. If O_i is a child of L_j, it has to be in at least one of L_j's children LPs. Similarly, O_i has to be in at least one of L_k's children LPs. If O_i appears in two (or more) lists, the algorithm joins the lists, and only a single LV is introduced. □

3.2 Algorithm correctness and complexity

Asymptotically, LPCC assures that the LGL-based LPCC algorithm learns a pattern of the true local MIM-based LVM [2], and assuming stationarity, this pattern is of the true temporal LVM even by using only a single time slice for learning. Lemma 2 shows that the algorithm is asymptotically correct, learning a pattern of the true temporal LVM regardless of the sequence size.

Lemma 2. *Assuming stationarity, the LGL-based LPCC algorithm is asymptotically correct, learning a pattern of the true temporal LVM.*

Proof. For each local graph, the LPCC algorithm learns a pattern of the true graph [2] if the data are infinite, and due to stationarity, all these local graphs are identical. Since the algorithm learns the most probable model (by a majority vote), then the global graph is identical to all local graphs that are a pattern of the true graph. □

With finite data, stationarity cannot guarantee correctness, and the algorithm learns the structure maximizing the a-posteriori probability of edges over local graphs under causal reasoning inferred using two orientation rules due to temporality (Sec. 3.1). Practically, probabilistic learning is implemented by a majority vote [16] for an edge, as described above, but can also be otherwise. Asymptotically, a majority vote is unnecessary (Lemma 2), but for the finite sample (leading to noise in the probabilistic connections among nodes), it improves performance as the number of voters (slices) increases [33] until the learned graph becomes probabilistically correct.

Voting over local graphs learned across time can also be represented as bagging [6]; in both cases, the samples are not independent (in our case, due to the overlap between the two-slice samples). This overlap between samples allows capturing significant temporal (and thus also causal) patterns in the data, e.g., in medical knowledge representation of chronic disease progression using longitudinal patient data. Our algorithm uses majority voting when transferring local graphs to the global one, and more specifically, when transferring local MSOs to global ones. Each OV is associated with the MSO maximizing the a-posteriori probability over all MSOs in the local graphs, and thereby with the most probable latent parent. For example, three of the local graphs [Figs. 2(a)–2(c)] vote to group variable X_4 with variables X_5 and X_6, while only one local graph votes for grouping it with X_1, X_2, and X_3. That is, each local graph $G_t \forall t$ is a weak learner to assign O_k to L_i. As the number of weak learners (sequence size) increases, the error margin [6]

$$mg(O_k, L_i) = av_t I(L(O_k^{G_t}) = L_i) - \max_{j \neq i} av_t I(L(O_k^{G_t}) = L_j),$$

converges (the probability that the margin is negative decreases), $I(\cdot)$ is the indicator function, and av_t is the average over local graphs.

The LPCC's first stage complexity is bounded by $O(|\mathbf{O}|^2 \cdot K^2 \cdot I)$, K and I are the numbers of clusters and iterations until convergence, respectively. LGL-based LPCC run-time is dictated by the LPCC's second stage, which estimates graph parameters in a single EM call (SEM-DBN, our algorithm counterpart, introduced in Sec. 4, needs many such calls). Since LGL-based LPCC is based on local learning that can run in parallel, combining local graphs adds only a minor contribution to the complexity, bounded by $O(|O|^2 \cdot (T-1)^2)$, T is the maximal sequence size. Run-time for our synthetic problems was minutes, while that of SEM-DBM was almost a day.

4 Empirical evaluation

We compared the LGL-based LPCC algorithm with two other algorithms that learn a latent DBN. First is SEM (this well-known algorithm [10, 24] is denoted here as SEM-DBN), which uses a search and score procedure to find the best fitted model from data, although not necessarily a causal one. Note that SEM-DBN requires the user to specify beforehand the number of LVs and their cardinalities. For fairness, we limited it: 1) to search over the (smaller) space of PMMs (we even initialized SEM-DBN with a random PMM), and 2) not to direct an edge from t to $t-1$. Second is tsFCI [9], which expands the FCI to time series and learns a causal temporal model that includes evidence for the existence of LVs (i.e., OVs connected by bi-directional edges imply on their parent LV), although it does not explicitly find them. To alleviate learning, we provided tsFCI with the values of the true LVs (as if they were observed), together with those of its OV children, which gives it a tremendous advantage (Table 4).

Table 4: Input fed to competitive algorithms.

	Number of LVs and their cardinality	LV values
tsFCI	Provided by the user	Provided by the user
SEM-DBN	Provided by the user	Learned from data
LGL-based LPCC	Learned from data	Learned from data

The algorithms were compared based on the structural Hamming distance (SHD) [32], which is the most common score for BN structure learning, counting structural differences (missing, reversed, and extra edges) between the true and learned graphs. Since the SHD cannot be applied directly to an LVM because the latent indices are arbitrary, we propose a new scoring method (Algorithm 2) which calculates the SHD for each combination of latent indices and adopts the minimal value as the score. Also, because SHD is a measure used to evaluate a directed acyclic graph, and tsFCI does not identify an LV directly but only implies on it by identifying a bi-directional edge between two of its OV children, tsFCI SHD-based error is huge, and to alleviate it, we provided tsFCI with the values of the true LVs, together with those of its OV children (Table 4). An additional new measure we suggest in Algorithm 2, *observed only SHD* (O-SHD), accounts only for latent to OV connections, and thereby evaluates the measurement model of the global LVM. The difference between the SHD and O-SHD measures is the error in learning the structural model, i.e., LV to LV relationships.

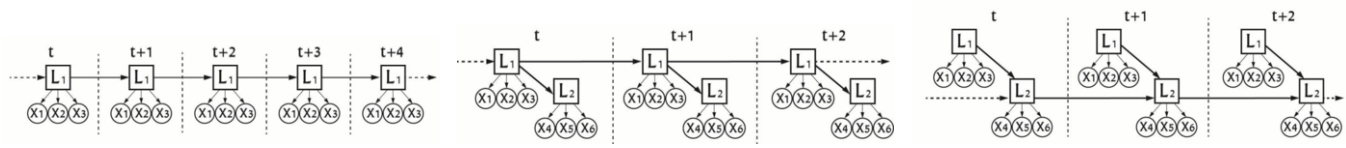
Input: *Learned PDAG H, True PDAG G, $|\mathbf{O}|$, $|\mathbf{L}|$*
Output: *Score (SHD), O-Score (O-SHD)*

C =all possible permutation of the vector: $[|\mathbf{O}|+1 : |\mathbf{O}|+|\mathbf{L}|]$;
Score=inf;
foreach *permutation in C* **do**
  i =The current LVs combination;
  $H' = H([1 : |\mathbf{O}|, i], [1 : |\mathbf{O}|, i])$;
  CurrentScore = call SHD procedure (H', G);
  if *CurrentScore ¡ Score* **then**
    Score = CurrentScore;
    O-Score = 0;
    foreach *edge E directed into an OV that is different between H' and G* **do**
      if *(E is missing in H') or (E is extra in H')* **then**
        O-Score = O-Score + 1;

Algorithm 2: Performance measures of LVM learning.

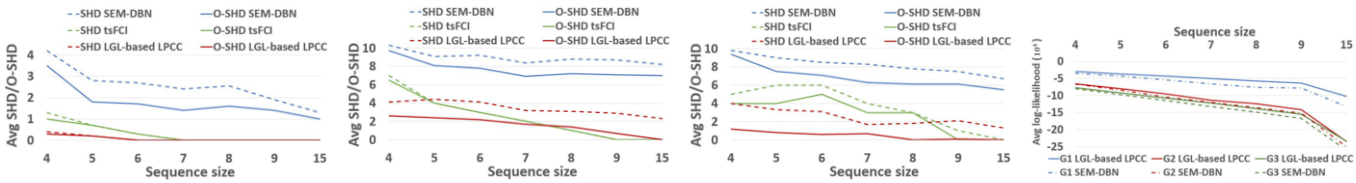
Using three artificial datasets, we first evaluated the three algorithms in structure learning (Sec. 4.1). The parameters of the models we synthesized represent temporal models. The datasets were sampled from three artificial BNs ($G1-G3$ in Fig. 3) of varying sequence sizes, $4 \leq T \leq 15$, i.e., a record is $(|\mathbf{O}| \times T)$-dimensional. The sample size was $D = \{2,000, 3,000, 4,000, 5,000, 10,000\}$. The cardinality of all variables was set to four, where $P(X_i = v | L_j = v) = 0.8$ and $P(X_i \neq v | L_j = v) = {}^{0.2}/_{3}$ (i.e., a 0.2 "noise" level was evenly distributed among the non-v values). These probabilities are the same for all T values to guarantee stationarity. Reported results are the modified SHD, O-SHD, and log-likelihood [14] averaged over ten data permutations for each value combination of G, T, and D.

The motivation to evaluate the LGL-based LPCC in classification (Sec. 4.2) and data imputation (Sec. 4.3) is twofold. First, these tasks allow us to appreciate the contribution of learning latent mechanisms,



(a) G1 (b) G2 (c) G3

Figure 3: Artificial temporal graphs with three OVs per LV and (a) a single LV, (b) two LVs, and (c) two LVs also making a collider per slice.



(a) $G1$ (b) $G2$ (c) $G3$ (d)

Figure 4: (a–c) SHD (dashed line) and O-SHD (solid line) in learning G1–G3 for different sequence sizes using SEM-DBN (blue), tsFCI (green), and LGL-based LPCC (red). Lower (error) values are better. (d) Log-likelihood in learning G1–G3 by the algorithms. Higher values are better.

which are at the heart of these tasks, and follow the improvement in the classification and data-imputation accuracies when introducing LVs compared to when ignoring them. Second, compared to experimentation with synthetic problems for which the true graph is known and scores of structural correctness (e.g., SHD) can easily be computed, in real-world (classification/imputation) problems, the true model is unknown; thus, the classification/imputation accuracy may provide a natural performance score of the LVM learning algorithm.

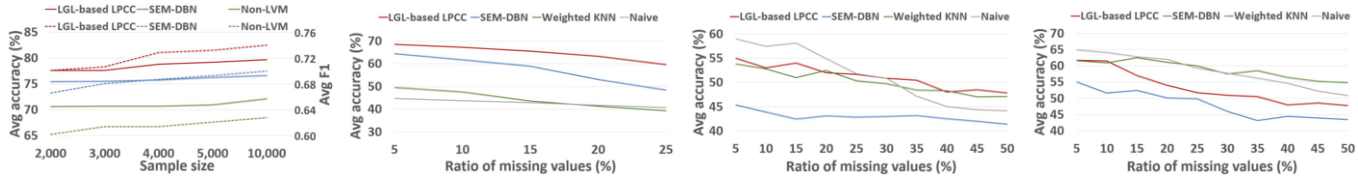
In Sec. 4.4, we compare the algorithms using real-world datasets of two neurodegenerative diseases and the imputation accuracy score.

4.1 Structure learning

Figs. 4(a)–4(c) show the average SHD and O-SHD, and Fig. 4(d) shows the log-likelihood for the three algorithms, 10,000 samples, and increasing sequence sizes. Since SHD measures errors in both learning the measurement and structural models, whereas O-SHD measures only the former, it is expected that the SHD scores will be higher than those of O-SHD. This is not the case for tsFCI (especially for $G1$ and $G2$ and less for $G3$, which is a more complex graph) where the algorithm identifies the structural model (latent to latent connections), but this is not surprising since it was fed with the LVs and only needed to find their relationships (Table 4). Although the LGL-based LPCC did not get this information, it did not fall behind tsFCI and was even superior in most cases. Fig. 4 reveals that the LGL-based LPCC and tsFCI are superior to the SEM-DBN regardless of the true graph, sequence size, and score. This superiority is statistically significant according to the non-parametric Wilcoxon signed ranks test [7] with a 0.05 confidence level regardless of G and the score. Fig. 4(a) shows that the LGL-based LPCC and tsFCI learn the true $G1$ graph accurately very quickly for a few sequences (with advantage to LGL-based LPCC), and even perfectly for $T \geq 6$ (LGL) or $T \geq 7$ (tsFCI). For $G2$ [Fig. 4(b)], the LGL-based LPCC learns the measurement model better and faster than both algorithms, but tsFCI learns the structural model better and faster (again, since it was provided with the LVs but not with their connections) and reaches a zero error a bit sooner. Although very small, the SHD error of the LGL-based LPCC for $G2$ and $G3$ is not zero because few latent–latent relationships are missed. This happens when the counts in matrix CO for an edge for both directions are equal, and thus, this edge is left undirected, contributing to the error. All performance measures of the SEM-DBN are inferior to the other algorithms, and this even includes the log-likelihood score [Fig. 4(d)], albeit the SEM-DBN is optimized by the log-likelihood [10, 24], and the other two are not. Finally, note that the LGL-based LPCC correctly learned the number of LVs for $G1$ and $G3$ for all sequence sizes, and for $G2$ for $T > 8$. This evaluation is irrelevant to the SEM-DBN and tsFCI that are already fed with the correct number of LVs.

4.2 Data classification

To demonstrate the contribution of learning a temporal LVM using the LGL-based LPCC algorithm to a classification task, we learned $G3$ [Fig. 3(c)], which represents a medical domain in which there is a (latent) cause to a disease, and we wish to predict the current disease state based on that cause and the previous disease state, where the cause and disease state each have three indicators (symptoms). We trained a random forest classifier for each of the six OVs of $G3$ in the Tth (last) slice (acting as the classification node, where all the remaining OVs in all slices are predictors), and averaged the classification accuracy over the six random forests to avoid any preference. We repeated this for each combination of sequence size, sample size, and data permutation. We compared this classification approach when LVs and their values were first identified using the LGL-based LPCC, were first identified using SEM-DBN, and when no latent variables exist (this model is denoted as non-LVM). In total, we trained $5 \times 8 \times 10 \times 6 \times 3 = 7,200$ classifiers for five sample sizes, eight sequence sizes, ten data permutations, six classifiers (classification nodes), and three models (LGL/SEM-DBN/no-LVM). We used 70% of the samples of each dataset for training and 30% for testing. Fig. 5(a) shows the average (over all data permutations, sequence sizes, and OVs) random-forest accuracy and F-measure for $G3$ and different sample sizes. It shows that the LGL-based LPCC achieves the best classification performance (significantly superior to the others), and that ignoring the existence of the LVs in the classification task results in poor performance. The LGL-based LPCC reaches the highest possible accuracy of 80% since the noise was originally set at 20%, i.e., even if the algorithm learns the true classification rule, in 20% of the cases, it will be wrong.



(a) Classification–$G3$ (b) Imputation–Synthetic (c) Imputation–ALS (d) Imputation–Alzheimer

Figure 5: a) Classification accuracy (solid line) and F-measure (dashed line) for increasing sample sizes, b–d) Imputation accuracy for increasing missing values for synthetic, ALS, and Alzheimer datasets, respectively.

4.3 Data imputation

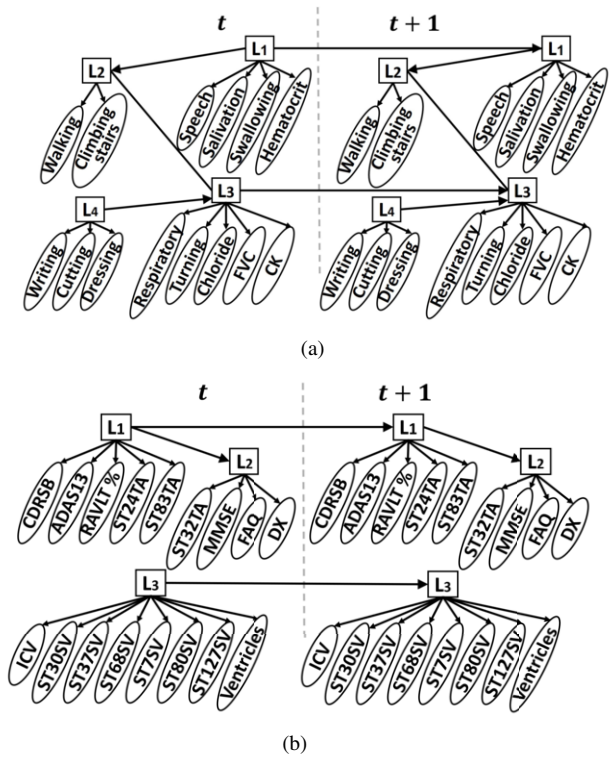
We sampled ten data permutations of 1,000 records each from $G3$ for each $4 \leq T \leq 15$ to make test sets to check the already learned LVMs. We randomly deleted 5–25% of the values in each test set (missing completely at random) and tried to complete the missing data based on: 1) an LGL-based LPCC; 2) an SEM-DBN, which is equivalent to an EM imputation [34]; 3) a weighted K-nearest neighbor [31]; and 4) a naïve method [5] in which a missing value is imputed based on the same variable in the previous slice (if not missing), the same variable in the next slice (if the previous one is missing and the next is not), or based on the variable's mean over all samples (if the variable in both the previous and next slices is missing). Fig. 5(b) shows that the LGL-based LPCC has the best imputation accuracy regardless of the missing value rate. The naïve approach performs better than random, but it is ranked last among the methods except for when the missing ratio is 25% and in which the weighted K-nearest neighbor is inferior. The SEM-DBN is more sensitive than the LGL-based LPCC to the missing value ratio because, as the ratio increases, the chance of two (or more) missing values occurring in a single latent's children within a sequence increases, which increases the importance of the parent latent (for which we have shown that the LGL-based LPCC is more accurate).

4.4 Real-world problems

Our amyotrophic lateral sclerosis (ALS) database consists of 3,171 patients with 22,089 clinic visits, from which we derived a subset of 2,590 patients having at least four visits, each two up to six months apart. A visit consists of lab test results and ten OVs measuring patient functionality in walking, writing, etc., each taking five values between 0 (a complete loss of function) and 4 (normal ability). The LGL-based LPCC learned a graph [Fig. 6(a)] that resembles medical categorization of patient functionality: bulbar functionality indicated mainly by speech, salivation, and swallowing; gross-motor functionality indicated by walking and climbing stairs; full body functionality indicated by respiratory ability, turning in bed, and three lab tests—chloride, FVC, and CK (also found correlated by [15]); and fine-motor functionality indicated by writing, cutting food, and dressing; (each group of functionalities is an MSO of a different latent: $L1$, $L2$, $L3$, and $L4$, respectively). The three intra-slice edges represent the natural connections between the bulbar and gross-motor, fine-motor and full body, and gross-motor and full body functionalities. The inter-slice edges between bulbar and full body to themselves complete the temporal-causal reasoning. Since the true graph was unknown, we could not report the SHD score, and instead, we evaluated the accuracy of the learned graph in data imputation (similar to Sec. 4.3).

Following the deletion of ALS data, Fig. 5(c) shows that the LGL-based LPCC outperforms the SEM-DBM with respect to imputation accuracy, regardless of the missing value ratio. The naïve method performs the best for small ratios of missingness, but as this ratio increases, the probability of a missing value in the previous/next observation increases and, thus, the method loses accuracy rapidly. Finally, the weighted K-nearest neighbor and LGL-based LPCC perform similarly, neither showing a significant advantage. This is encouraging since the former is a state-of-the-art imputation method, and the latter is a temporal LVM learning algorithm that we decided to check in an imputation task.

The Alzheimer's data mainly contain variables derived from processed radiological images of 1,737 patients with 12,741 clinic visits. Fig. 6(b) shows the resultant graph with a connection between L1 and L2 that represent cognitive ability, as is measured by different cognitive tests (e.g., ADAS13, RAVLT, MMSE, FAQ), and a connection between L3 to itself (inter-slice edge) that may represent deterioration in patient physiological ability, as indicated by normalized ventricles and radiological features. Similar to the ALS, in the absence of the true graph, we report on data imputation accuracies. Fig. 5(d) shows that the dedicated imputation methods (naïve and weighted K-nearest neighbor) outperform the (non-dedicated to imputation) LGL-based LPCC and SEM-DBN algorithms in the presence of missing data.



(a)

(b)

Figure 6: Temporal LVMs learned by the LGL-based LPCC for a) the ALS database and b) the Alzheimer database.

This is because the LVs in the learned graph [Fig. 6(b)] are not all connected; thus, missing values of L_3 indicators, for example, cannot be estimated using the indicators of L_1 and L_2. Nevertheless, the LGL-based LPCC still outperforms the SEM-DBM with respect to imputation accuracy, regardless of the missing value ratio.

5 Summary

Causal discovery in the presence of LVs is a difficult problem, becoming even more challenging when it is temporal. Models by most learning algorithms of a latent DBN may fit the data well and even demonstrate evidence of latent mechanisms, but without showing causal relations. The LGL-based LPCC algorithm is a causal discovery mechanism to learn a latent DBN based on PMMs learned locally for each 2-TBN. This algorithm showed clear superiority over the state-of-the-art SEM-DBN and tsFCI algorithms with respect to structural correctness, fitting the data, accuracy in classification and imputation, and run-time even though we provided the SEM-DBN with the correct number of LVs and their cardinalities and the tsFCI with the LVs themselves. Although the PMM assumption on which LGL-based LPCC is based upon is commonly used (to limit the structure space and enable practical learning) and justified [27, 23], it is restrictive, and we are exploring ways to expand the LPCC theory to learn a non-PMM or to apply post-processing to modify the learned graph into a non-PMM. Finally, the code for the algorithm and the datasets used in this work are available at http://www.ee.bgu.ac.il/ boaz.

REFERENCES

[1] C.F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X.D. Koutsoukos, 'Local causal and Markov blanket induction for causal discovery and feature selection for classification Part II: Analysis and extensions', *Journal of Machine Learning Research*, **11**, 235–284, (2010).

[2] N. Asbeh and B. Lerner, 'Learning latent variable models by pairwise cluster comparison: Part I–Theory and overview', *Journal of Machine Learning Research*, **17**(224), 1–52, (2016).

[3] N. Asbeh and B. Lerner, 'Learning latent variable models by pairwise cluster comparison: Part II–Algorithm and evaluation', *Journal of Machine Learning Research*, **17**(233), 1–45, (2016).

[4] F.J. Bartholomew, J.I. Galbraith, I. Moustaki, and F. Steele, *The analysis and interpretation of multivariate data for social scientists*, CRC Press, 2002.

[5] G.E Batista and M.C. Monard, 'An analysis of four missing data treatment methods for supervised learning', *Applied Artificial Intelligence*, **17**(5-6), 519–533, (2003).

[6] L. Breiman, 'Random forests', *Machine Learning*, **45**(1), 5–32, (2001).

[7] J. Demsar, 'Statistical comparisons of classifiers over multiple data sets', *Journal of Machine Learning Research*, **7**(1), 1–30, (2006).

[8] G. Elidan, N. Lotner, N. Friedman, and D. Koller, 'Discovering hidden variables: A structure-based approach', in *Advances in Neural Information Processing Systems*, pp. 479–485, (2000).

[9] D. Entner and P.O. Hoyer, 'On causal discovery from time series data using FCI', *Probabilistic graphical models*, 121–128, (2010).

[10] N. Friedman, 'The Bayesian structural EM algorithm', in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp. 129–138. Morgan Kaufmann Publishers Inc., (1998).

[11] A. Garg, V. Pavlovic, and J.M. Rehg, 'Audio-visual speaker detection using dynamic Bayesian networks', in *Proceedings 4th International Conference on Automatic Face and Gesture Recognition*, pp. 384–390. IEEE, (2000).

[12] Z. Ghahramani and M.I. Jordan, 'Factorial hidden Markov models', in *Advances in Neural Information Processing Systems*, pp. 472–478, (1996).

[13] S. Harmeling and C.K.I. Williams, 'Greedy learning of binary latent trees', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **33**(6), 1087–1097, (2011).

[14] D. Heckerman, D. Geiger, and D.M. Chickering, 'Learning Bayesian networks: the combination of knowledge and statistical data', *Machine Learning*, **20**(3), 197–243, (1995).

[15] K. Ikeda, T. Hirayama, T. Takazawa, K. Kawabe, and Y. Iwasaki, 'Relationships between disease progression and serum levels of lipid, urate, creatinine and ferritin in Japanese patients with amyotrophic lateral sclerosis: a cross-sectional study', *Internal Medicine*, **51**(12), 1501–1508, (2012).

[16] G. James, *Majority vote classifiers: Theory and applications*, Ph.D. dissertation, Stanford University, 1998.

[17] R. Klee, *Introduction to the philosophy of science: cutting nature at its seams*, Oxford University Press, New York, New York, 1997.

[18] E. Kummerfeld and J. Ramsey, 'Causal clustering for 1-factor measurement models', in *Proceedings of the 22nd ACM SIGKDD*, pp. 1655–1664. ACM, (2016).

[19] E. Kummerfeld, J. Ramsey, R. Yang, P. Spirtes, and R. Scheines, 'Causal clustering for 2-factor measurement models', in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 34–49. Springer, (2014).

[20] D. Lewis, 'Counterfactual dependence and time's arrow', *Noûs*, 455–476, (1979).

[21] D. Margaritis and S. Thrun, 'Bayesian network induction via local neighborhoods', in *Advances in Neural Information Processing Systems*, pp. 505–511, (1999).

[22] V. Mihajlovic and M. Petkovic, 'Dynamic Bayesian networks: A state of the art', Technical report, University of Twente Document Repository, (2001).

[23] R. Mourad, C. Sinoquet, N.L Zhang, T. Liu, and P. Leray, 'A survey on latent tree models and applications', *Journal of Artificial Intelligence Research*, **47**, 157–203, (2013).

[24] K.P. Murphy, *Machine learning: A probabilistic perspective*, MIT press, 2014.

[25] L.R. Rabiner, 'A tutorial on hidden Markov models and selected applications in speech recognition', *Proceedings of the IEEE*, **77**(2), 257–286, (1989).

[26] J.W. Robinson and A.J. Hartemink, 'Learning non-stationary dynamic bayesian networks', *Journal of Machine Learning Research*, **11**(Dec), 3647–3680, (2010).

[27] R. Silva, C. Glymour, and P. Spirtes, 'Learning the structure of linear latent variable models', *Journal of Machine Learning Research*, **7**, 191–246, (2006).

[28] L. Song, M. Kolar, and E.P. Xing, 'Time-varying dynamic Bayesian networks', in *Advances in Neural Information Processing Systems*, pp. 1732–1740, (2009).

[29] P. Spirtes, C.N. Glymour, and R. Scheines, *Causation, prediction, and search*, MIT press, 2000.

[30] G. Trabelsi, P. Leray, M.B. Ayed, and A.M. Alimi, 'Benchmarking dynamic Bayesian network structure learning algorithms', in *Proceedings 5th International Conference on Modeling, Simulation and Applied Optimization*, pp. 1–6. IEEE, (2013).

[31] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R.B. Altman, 'Missing value estimation methods for dna microarrays', *Bioinformatics*, **17**(6), 520–525, (2001).

[32] I. Tsamardinos, L.E. Brown, and C.F. Aliferis, 'The max-min hill-climbing Bayesian network structure learning algorithm', *Machine Learning*, **65**(1), 31–78, (2006).

[33] G. Valentini and F. Masulli, 'Ensembles of learning machines', in *Italian Workshop on Neural Nets*, pp. 3–20. Springer, (2002).

[34] C. Yozgatligil, S. Aslan, C. Iyigun, and I. Batmaz, 'Comparison of missing value imputation methods in time series: the case of turkish meteorological data', *Theoretical and Applied Climatology*, **112**(1-2), 143–167, (2013).

[35] N.L. Zhang, 'Hierarchical latent class models for cluster analysis', *Journal of Machine Learning Research*, **5**(6), 697–723, (2004).

[36] N.L. Zhang, W. Yi, and C. Tao, 'Discovery of latent structures: Experience with the coil challenge 2000 data set', *Journal of Systems Science and Complexity*, **21**(2), 172–183, (2008).