Journal of Machine Learning Research 12 (2011) 863-905 Submitted 10/09; Revised 10/10; Published 3/11

Sparse Linear Identifiable Multivariate Modeling

Ricardo Henao[*] RHENAO@BINF.KU.DK
Ole Winther[*] OWI@IMM.DTU.DK
DTU Informatics
Richard Petersens Plads, Building 321
Technical University of Denmark
DK-2800 Lyngby, Denmark

Editor: Aapo Hyvärinen

Abstract

In this paper we consider sparse and identifiable linear latent variable (factor) and linear Bayesian network models for parsimonious analysis of multivariate data. We propose a computationally efficient method for joint parameter and model inference, and model comparison. It consists of a fully Bayesian hierarchy for sparse models using slab and spike priors (two-component δ-function and continuous mixtures), non-Gaussian latent factors and a stochastic search over the ordering of the variables. The framework, which we call SLIM (Sparse Linear Identifiable Multivariate modeling), is validated and bench-marked on artificial and real biological data sets. SLIM is closest in spirit to LiNGAM (Shimizu et al., 2006), but differs substantially in inference, Bayesian network structure learning and model comparison. Experimentally, SLIM performs equally well or better than LiNGAM with comparable computational complexity. We attribute this mainly to the stochastic search strategy used, and to parsimony (sparsity and identifiability), which is an explicit part of the model. We propose two extensions to the basic i.i.d. linear framework: non-linear dependence on observed variables, called SNIM (Sparse Non-linear Identifiable Multivariate modeling) and allowing for correlations between latent variables, called CSLIM (Correlated SLIM), for the temporal and/or spatial data. The source code and scripts are available from `http://cogsys.imm.dtu.dk/slim/`.

Keywords: parsimony, sparsity, identifiability, factor models, linear Bayesian networks

1. Introduction

Modeling and interpretation of multivariate data are central themes in machine learning. Linear latent variable models (or factor analysis) and linear directed acyclic graphs (DAGs) are prominent examples of models for continuous multivariate data. In factor analysis, data is modeled as a linear combination of independently distributed factors thus allowing for capture of a rich underlying co-variation structure. In the DAG model, each variable is expressed as regression on a subset of the remaining variables with the constraint that total connectivity is acyclic in order to have a properly defined joint distribution. Parsimonious (interpretable) modeling, using sparse factor loading matrix or restricting the number of parents of a node in a DAG, are good prior assumptions in many applications. Recently, there has been a great deal of interest in detailed modeling of sparsity in factor models, for example in the context of gene expression data analysis (West, 2003; Lucas et al.,

[*] Ricardo Henao and Ole Winther hold joint affiliations with DTU Informatics, Technical University of Denmark and the Bioinformatics Centre, University of Copenhagen.

2006; Knowles and Ghahramani, 2007; Thibaux and Jordan, 2007; Carvalho et al., 2008; Rai and Daume III, 2009). Sparsity arises for example in gene regulation because the latent factors represent driving signals for gene regulatory sub-networks and/or transcription factors, each of which only includes/affects a limited number of genes. A parsimonious DAG is particularly attractable from an interpretation point of view but the restriction to only having observed variables in the model may be a limitation because one rarely measures all relevant variables. Furthermore, linear relationships might be unrealistic for example in gene regulation, where it is generally accepted that one cannot replace the driving signal (related to concentration of a transcription factor protein in the cell nucleus) with the measured concentration of corresponding mRNA. Bayesian networks represent a very general class of models, encompassing both observed and latent variables. In many situations it will thus be relevant to learn parsimonious Bayesian networks with both latent variables and a non-linear DAG parts. Although attractive, by being closer to what one may expect in practice, such modeling is complicated by difficult inference (Chickering 1996 showed that DAG structure learning is NP-hard) and by potential non-identifiability. Identifiability means that each setting of the parameters defines a unique distribution of the data. Clearly, if the model is not identifiable in the DAG and latent parameters, this severely limits the interpretability of the learned model.

Shimizu et al. (2006) provided the important insight that every DAG has a factor model representation, that is, the connectivity matrix of a DAG gives rise to a triangular mixing matrix in the factor model. This provided the motivation for the Linear Non-Gaussian Acyclic Model (LiNGAM) algorithm which solves the identifiable factor model using Independent Component Analysis (ICA, Hyvärinen et al., 2001) followed by iterative permutation of the solutions towards triangular, aiming to find a suitable ordering for the variables. As final step, the resulting DAG is pruned based on different statistics, for example, Wald, Bonferroni, χ^2 second order model fit tests. Model selection is then performed using some pre-chosen significance level, thus LiNGAM select from models with different sparsity levels and a fixed deterministically found ordering. There is a possible number of extensions to their basic model, for instance Hoyer et al. (2008) extend it to allow for latent variables, for which they use a probabilistic version of ICA to obtain the variable ordering, pruning to make the model sparse and bootstrapping for model selection. Although the model seems to work well in practice, as commented by the authors, it is restricted to very small problems (3 or 4 observed and 1 latent variables). Non-linear DAGs are also a possibility, however finding variable orderings in this case is known to be far more difficult than in the linear case. These methods inspired by Friedman and Nachman (2000), mainly consist of two steps: performing non-linear regression for a set of possible orderings, and then testing for independence to prune the model, see for instance Hoyer et al. (2009) and Zhang and Hyvärinen (2010). For tasks where exhaustive order enumeration is not feasible, greedy approaches like DAG-search (see “ideal parent” algorithm, Elidan et al., 2007) or PC (Prototypical Constraint, see kernel PC, Tillman et al., 2009) can be used as computationally affordable alternatives.

Factor models have been successfully employed as exploratory tools in many multivariate analysis applications. However, interpretability using sparsity is usually not part of the model, but achieved through post-processing. Examples of this include, bootstrapping, rotating the solutions to maximize sparsity (varimax, procrustes), pruning or thresholding. Another possibility is to impose sparsity in the model through L_1 regularization to obtain a maximum a-posteriori estimate (Jolliffe et al., 2003; Zou et al., 2006). In fully Bayesian sparse factor modeling, two approaches have been proposed: parametric models with bimodal sparsity promoting priors (West, 2003; Lucas et al., 2006; Carvalho et al., 2008; Henao and Winther, 2009), and non-parametric models where the number of factors is potentially infinite (Knowles and Ghahramani, 2007; Thibaux and Jordan, 2007; Rai and Daume III, 2009). It turns out that most of the parametric sparse factor models can be seen as finite versions of their non-parametric counterparts, for instance West (2003) and Knowles and Ghahramani (2007). The model proposed by West (2003) is, as far as the authors know, the first attempt to encode sparsity in a factor model explicitly in the form of a prior. The remaining models improve the initial setting by dealing with the optimal number of factors in Knowles and Ghahramani (2007), improved hierarchical specification of the sparsity prior in Lucas et al. (2006), Carvalho et al. (2008) and Thibaux and Jordan (2007), hierarchical structure for the loading matrices in Rai and Daume III (2009) and identifiability without restricting the model in Henao and Winther (2009).

Many algorithms have been proposed to deal with the NP-hard DAG structure learning task. LiNGAM, discussed above, is the first fully identifiable approach for continuous data. All other approaches for continuous data use linearity and (at least implicitly) Gaussianity assumptions so that the model structure learned is only defined up to equivalence classes. Thus in most cases the directionality information about the edges in the graph must be discarded. Linear Gaussian-based models have the added advantage that they are computationally affordable for the many variables case. The structure learning approaches can be roughly divided into stochastic search and score (Cooper and Herskovits, 1992; Heckerman et al., 2000; Friedman and Koller, 2003), constraint-based (with conditional independence tests) (Spirtes et al., 2001) and two stage; like LiNGAM, (Tsamardinos et al., 2006; Friedman et al., 1999; Teyssier and Koller, 2005; Schmidt et al., 2007; Shimizu et al., 2006). In the following, we discuss in more detail previous work in the last category, as it is closest to the work in this paper and can be considered representative of the state-of-the-art. The Max-Min Hill-Climbing algorithm (MMHC, Tsamardinos et al., 2006) first learns the skeleton using conditional independence tests similar to PC algorithms (Spirtes et al., 2001) and then the order of the variables is found using a Bayesian-scoring hill-climbing search. The Sparse Candidate (SC) algorithm (Friedman et al., 1999) is in the same spirit but restricts the skeleton to within a predetermined link candidate set of bounded size for each variable. The Order Search algorithm (Teyssier and Koller, 2005) uses hill-climbing first to find the ordering, and then looks for the skeleton with SC. L_1 regularized Markov Blanket (Schmidt et al., 2007) replaces the skeleton learning from MMHC with a dependency network (Heckerman et al., 2000) written as a set of local conditional distributions represented as regularized linear regressors. Since the source of identifiability in Gaussian DAG models is the direction of the edges in the graph, a still meaningful approach consists of entirely focusing on inferring the skeleton of the graph by keeping the edges undirected as in Dempster (1972), Dawid and Lauritzen (1993), Giudici and Green (1999) and Rajaratman et al. (2008).

In this paper we propose a framework called SLIM (Sparse Linear Identifiable Multivariate modeling, see Figure 1) in which we learn models from a rather general class of Bayesian networks and perform quantitative model comparison between them.[1] Model comparison may be used for model selection or serve as a hypothesis-generating tool. We use the likelihood on a test set as a computationally simple quantitative proxy for model comparison and as an alternative to the marginal likelihood. The other two key ingredients in the framework are the use of sparse and identifiable model components (Carvalho et al., 2008; Kagan et al., 1973, respectively) and the stochastic search for the correct order of the variables needed by the DAG representation. Like LiNGAM, SLIM ex-

[1] A preliminary version of our approach appears in NIPS 2009: Henao and Winther, Bayesian sparse factor models and DAGs inference and comparison.

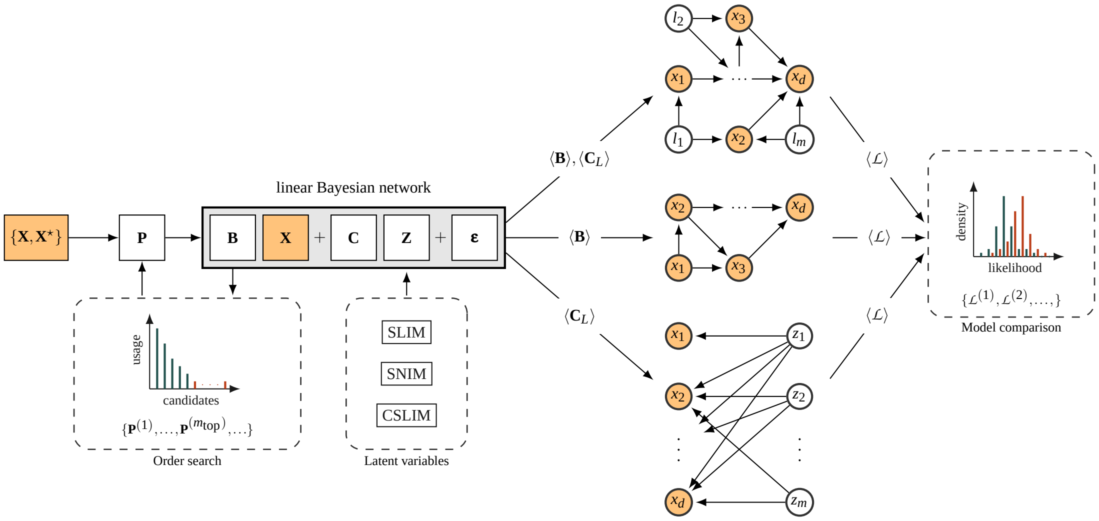

$\{\mathbf{X},\mathbf{X}^\star\}$
P
linear Bayesian network
B
X
+
C
Z
+
ε
usage
candidates
$\{\mathbf{P}^{(1)},\ldots,\mathbf{P}^{(m_{\text{top}})},\ldots\}$
Order search
SLIM
SNIM
CSLIM
Latent variables
$\langle\mathbf{B}\rangle,\langle\mathbf{C}_L\rangle$
$\langle\mathbf{B}\rangle$
$\langle\mathbf{C}_L\rangle$
l_2
x_3
x_1
$\cdots$
x_d
l_1
x_2
l_m
x_2
$\cdots$
x_d
x_1
x_3
x_1
z_1
x_2
z_2
x_d
z_m
$\langle\mathcal{L}\rangle$
$\langle\mathcal{L}\rangle$
$\langle\mathcal{L}\rangle$
density
likelihood
$\{\mathcal{L}^{(1)},\mathcal{L}^{(2)},\ldots,\}$
Model comparison

Figure 1: SLIM in a nutshell. Starting from a training-test set partition of data $\{\mathbf{X},\mathbf{X}^\star\}$, our framework produces factor models $\mathbf{C}$ and DAG candidates $\mathbf{B}$ with and without latent variables $\mathbf{Z}$ that can be compared in terms of how well they fit the data using test likelihoods $\mathcal{L}$. The variable ordering $\mathbf{P}$ needed by the DAG is obtained as a byproduct of a factor model inference. Besides, changing the prior over latent variables $\mathbf{Z}$ produces two variants of SLIM called CSLIM and SNIM.

ploits the close relationship between factor models and DAGs. However, since we are interested in the factor model by itself, we will not constrain the factor loading matrix to have triangular form, but allow for sparse solutions so pruning is not needed. Rather we may ask whether there exists a permutation of the factor-loading matrix agreeing to the DAG assumption (in a probabilistic sense). The slab and spike prior biases towards sparsity so it makes sense to search for a permutation in parallel with factor model inference. We propose to use stochastic updates for the permutation using a Metropolis-Hastings acceptance ratio based on likelihoods with the factor-loading matrix being masked. In practice this approach gives good solutions up to at least fifty dimensions. Given a set of possible variable orderings inferred by this method, we can then learn DAGs using slab and spike priors for their connectivity matrices. The so-called slab and spike prior is a two-component mixture of a continuous distribution and degenerate δ-function point mass at zero. This type of model implicitly defines a prior over structures and is thus a computationally attractive alternative to combinatorial structure search since parameter and structure inference are performed simultaneously. A key to effective learning in these intractable models is Markov Chain Monte Carlo (MCMC) sampling schemes that mix well. For non-Gaussian heavy-tailed distributions like the Laplace and t-distributions, Gibbs sampling can be efficiently defined using appropriate infinite scale mixture representations of these distributions (Andrews and Mallows, 1974). We also show that our model is very flexible in the sense that it can be easily extended by only changing the prior distribution of a set of latent variables, for instance to allow for time series data (CSLIM, Correlated SLIM) and non-linearities in the DAG structure (SNIM, Sparse non-Linear Identifiable Multivariate modeling) through Gaussian process priors.

The rest of the paper is organized as follows: Section 2 describes the model and its identifiability properties. Section 3 provides all prior specification including sparsity, latent variables and driving signals, order search and extensions for correlated data (CSLIM) and non-linearities (SNIM). Section 4 elaborates on model comparison. Section 5 and Appendix A provide an overview of the model and practical details on the MCMC-based inference, proposed workflow and computational cost requirements. Section 6 contains the experiments. We show simulations based on artificial data to illustrate all the features of the model proposed. Real biological data experiments illustrate the advantages of considering different variants of Bayesian networks. For all data sets we compare with some of the most relevant existing methods. Section 7 concludes with a discussion, open questions and future directions.

2. Linear Bayesian Networks

A Bayesian network is essentially a joint probability distribution defined via a directed acyclic graph, where each node in the graph represents a random variable x. Due to the acyclic property of the graph, its node set $x_1,\dots,x_d$ can be partitioned into d subsets $\{V_1,V_2,\dots,V_d\} \equiv \mathcal{V}$, such that if $x_j \to x_i$ then $x_j \in V_i$, that is, V_i contains all *parents* of x_i. We can then write the joint distribution as a product of conditionals of the form

$$P(x_1,\dots,x_d) = \prod_{i=1}^{d} P(x_i|V_i) \ ,$$

thus x_i is conditionally independent of $\{x_j|x_i \notin V_j\}$ given V_i for $i \neq j$. This means that $p(x_1,\dots,x_d)$ can be used to describe the joint probability of any set of variables once $\mathcal{V}$ is given. The problem is that $\mathcal{V}$ is usually unknown and thus needs to be (at least partially) inferred from observed data.

We consider a model for a fairly general class of linear Bayesian networks by putting together a linear DAG, $\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{z}$, and a factor model, $\mathbf{x} = \mathbf{C}\mathbf{z} + \boldsymbol{\varepsilon}$. Our goal is to explain each one of d observed variables $\mathbf{x}$ as a linear combination of the remaining ones, a set of $d+m$ independent latent variables $\mathbf{z}$ and additive noise $\boldsymbol{\varepsilon}$. We have then

$$\mathbf{x} = (\mathbf{R} \odot \mathbf{B})\mathbf{x} + (\mathbf{Q} \odot \mathbf{C})\mathbf{z} + \boldsymbol{\varepsilon} \ , \tag{1}$$

where $\odot$ is the element-wise product and we can further identify the following elements:

- $\mathbf{z}$ is partitioned into two subsets, $\mathbf{z}_D$ is a set of d driving signals for each observed variable in $\mathbf{x}$ and $\mathbf{z}_L$ is a set of m shared general purpose latent variables. $\mathbf{z}_D$ is used here to describe the intrinsic behavior of the observed variables that cannot regarded as "external" noise.

- $\mathbf{R}$ is a $d \times d$ binary connectivity matrix that encodes whether there is an edge between observed variables, by means of $r_{ij} = 1$ if $x_i \to x_j$. Since every non-zero element in $\mathbf{R}$ is an edge of a DAG, $r_{ii} = 0$ and $r_{ij} = 0$ if $r_{ji} \neq 0$ to avoid self-interactions and bi-directional edges, respectively. This also implies that there is at least one permutation matrix $\mathbf{P}$ such that $\mathbf{P}^\top \mathbf{R}\mathbf{P}$ is strictly lower triangular where we have used that $\mathbf{P}$ is orthonormal then $\mathbf{P}^{-1} = \mathbf{P}^\top$.

- $\mathbf{Q} = [\mathbf{Q}_D \ \mathbf{Q}_L]$ is a $d \times (d+m)$ binary connectivity matrix, this time for the conditional independence relations between observed and latent variables. We assume that each observed variable has a dedicated latent variable, thus the first d columns of $\mathbf{Q}_D$ are the identity. The remaining m columns can be arbitrarily specified, by means of $q_{ij} \neq 0$ if there is an edge between x_i and z_j for $d < j \leq m$.

- $\mathbf{B}$ and $\mathbf{C} = [\mathbf{C}_L\ \mathbf{C}_D]$ are respectively, $d \times d$ and $d \times (d+m)$ weight matrices containing the edge strengths for the Bayesian network. Their elements are constrained to be non-zero only if their corresponding connectivities are also non-zero.

The model (1) has two important special cases, (i) if all elements in $\mathbf{R}$ and $\mathbf{Q}_D$ are zero it becomes a standard factor model (FM) and (ii) if $m = 0$ or all elements in $\mathbf{Q}_L$ are zero it is a pure DAG. The model is not a completely general linear Bayesian network because connections to latent variables are absent (see for example Silva, 2010). However, this restriction is mainly introduced to avoid compromising the identifiability of the model. In the following we will only write $\mathbf{Q}$ and $\mathbf{R}$ explicitly when we specify the sparsity modeling.

2.1 Identifiability

We will split the identifiability of the model in Equation (1) in three parts addressing first the factor model, second the pure DAG and finally the full model. By identifiability we mean that each different setting of the parameters $\mathbf{B}$ and $\mathbf{C}$ gives a unique distribution of the data. In some cases the model is only unique up to some symmetry of the model. We discuss these symmetries and their effect on model interpretation in the following.

Identifiability in factor models $\mathbf{x} = \mathbf{C}_L\mathbf{z}_L + \boldsymbol{\varepsilon}$ can be obtained in a number of ways (see Chapter 10, Kagan et al., 1973). Probably the easiest way is to assume sparsity in $\mathbf{C}_L$ and restrict its number of free parameters, for example by restricting the dimensionality of $\mathbf{z}$, namely m, according to the Ledermann bound $m \leq (2d+1-(8d+1)^{1/2})/2$ (Bekker and ten Berge, 1997). The Ledermann bound guarantees the identification of $\boldsymbol{\varepsilon}$ and follows just from counting the number of free parameters in the covariance matrices of $\mathbf{x}$, $\boldsymbol{\varepsilon}$ and in $\mathbf{C}_L$, assuming Gaussianity of $\mathbf{z}$ and $\boldsymbol{\varepsilon}$. Alternatively, identifiability is achieved using non-Gaussian distributions for $\mathbf{z}$. Kagan et al. (Theorem 10.4.1, 1973) states that when at least $m-1$ latent variables are non-Gaussian, $\mathbf{C}_L$ is identifiable up to scale and permutation of its columns, that is, we can identify $\widehat{\mathbf{C}}_L = \mathbf{C}_L\mathbf{S}_\mathrm{f}\mathbf{P}_\mathrm{f}$, where $\mathbf{S}_\mathrm{f}$ and $\mathbf{P}_\mathrm{f}$ are arbitrary scaling and permutation matrices, respectively. Comon (1994) provided an alternative well-known proof for the particular case of $m-1=d$. The $\mathbf{S}_\mathrm{f}$ and $\mathbf{P}_\mathrm{f}$ symmetries are inherent in the factor model definition in all cases and will usually not affect interpretability. However, some researchers prefer to make the model completely identifiable, for example, by making $\mathbf{C}_L$ triangular with non-negative diagonal elements (Lopes and West, 2004). In addition, if all components of $\boldsymbol{\varepsilon}$ are Gaussian and the rank of $\mathbf{C}_L$ is m, then the distributions of $\mathbf{z}$ and $\boldsymbol{\varepsilon}$ are uniquely defined to within common shift in mean (Theorem 10.4.3, Kagan et al., 1973). In this paper, we use the non-Gaussian $\mathbf{z}$ option for two reasons, (i) restricting the number of latent variables severely limits the usability of the model and (ii) non-Gaussianity is a more realistic assumption in many application areas such as for example biology.

For pure DAG models $\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{C}_D\mathbf{z}_D$, identifiability can be obtained using the factor model result from Kagan et al. (1973) by rewriting the DAG into an equivalent factor model $\mathbf{x} = \mathbf{D}\mathbf{z}$ with $\mathbf{D} = (\mathbf{I}-\mathbf{B})^{-1}\mathbf{C}_D$, see Figure 2. From the factor model result it only follows that $\mathbf{D}$ is identifiable up to a scaling and permutation. However, as mentioned above, due to the acyclicity there is at least one permutation matrix $\mathbf{P}$ such that $\mathbf{P}^\top\mathbf{B}\mathbf{P}$ is strictly lower triangular. Now, if $\mathbf{x}$ admits DAG representation, the same $\mathbf{P}$ makes the permuted $\widehat{\mathbf{D}} = (\mathbf{I}-\mathbf{P}^\top\mathbf{B}\mathbf{P})^{-1}\mathbf{C}_D$, triangular with $\mathbf{C}_D$ on its diagonal. The constraint on the number of non-zero elements in $\mathbf{D}$ due to triangularity removes the permutation freedom $\mathbf{P}_\mathrm{f}$ such that we can subsequently identify $\mathbf{P}$, $\mathbf{B}$ and $\mathbf{C}_D$. It also implies that any valid permutation $\mathbf{P}$ will produce exactly the same distribution for $\mathbf{x}$.

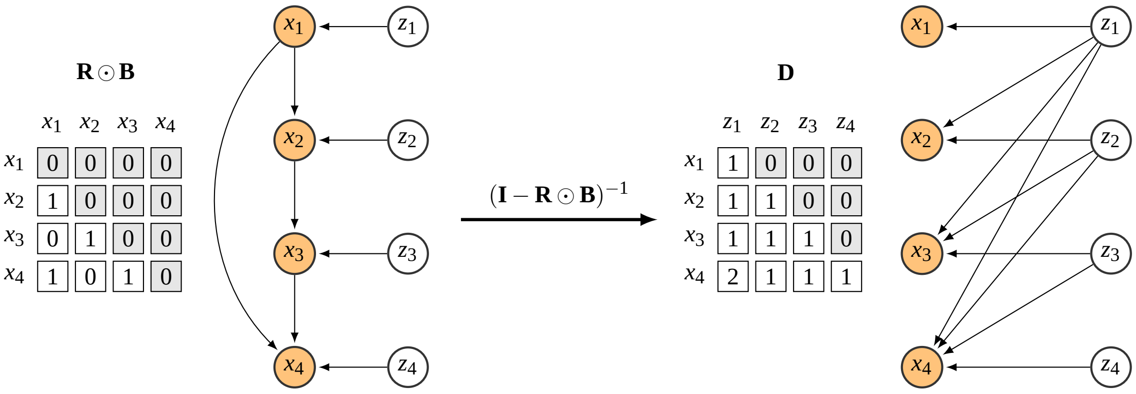


Figure 2: FM-DAG equivalence illustration. In the left side, a DAG model with four variables with corresponding connectivity matrix $\mathbf{R}$, $b_{ij} = 1$ when $r_{ij} = 1$ and $\mathbf{C}_D = \mathbf{I}$. In the right hand side, the equivalent factor model with mixing matrix $\mathbf{D}$. Note that the factor model is sparse even if its corresponding DAG is dense. The gray boxes in $\mathbf{D}$ and $\mathbf{R} \odot \mathbf{B}$ represent elements that must be zero by construction.

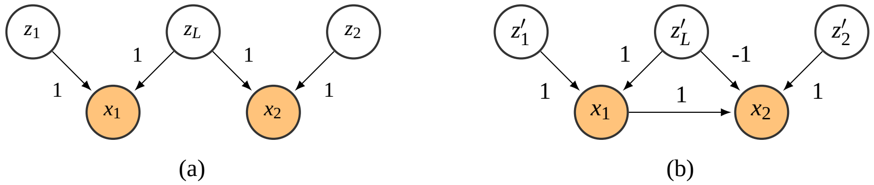
In the general case in Equation (1), $\mathbf{D} = (\mathbf{I} - \mathbf{B})^{-1}\mathbf{C}$ is of size $d \times (d+m)$. What we will show is that even if $\mathbf{D}$ is still identifiable, we can no longer obtain $\mathbf{B}$ and $\mathbf{C}$ uniquely unless we "tag" the model by requiring the distributions of driving signals $\mathbf{z}_D$ and latent signals $\mathbf{z}_L$ to differ. In order to illustrate why we get non-identifiability, we can write $\mathbf{x} = \mathbf{D}\mathbf{z}$ inverting $\mathbf{D}$ explicitly. For simplicity we consider $m = 1$ and $\mathbf{P} = \mathbf{I}$ but generalizing to $m > 1$ is straight forward

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_d \end{bmatrix} = \begin{bmatrix} c_{11} & 0 & 0 & \cdots & c_{1L} \\ b_{21}c_{11} & c_{22} & 0 & \cdots & b_{21}c_{1L} + c_{2L} \\ b_{31}c_{11} + b_{32}b_{21}c_{11} & b_{32}c_{22} & c_{33} & \cdots & b_{31}c_{1L} + b_{32}b_{21}c_{1L} + a_{32}c_{2L} + c_{3L} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{11} + \sum_{k=1}^{i-1} b_{ik}d_{k1} & \cdots & \cdots & \cdots & c_{iL} + \sum_{k=1}^{i-1} b_{ik}d_{kL} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{d+1} \end{bmatrix} .$$

We see from this equation that if all latent variables have the same distribution and c_{1L} is non-zero then we may exchange the first and last column in $\mathbf{D}$ to get two equivalent distributions with different elements for $\mathbf{B}$ and $\mathbf{C}$. The model is thus non-identifiable. If the first i elements in latent column of $\mathbf{C}$ are zero then the $(i+1)$-th and last column can be exchanged. Hoyer et al. (2008) made the same basic observation through a number of examples. Interestingly, we also see from the triangularity requirement of the "driving signal" part of $\mathbf{D}$ that $\mathbf{P}$ is actually identifiable despite the fact that $\mathbf{B}$ and $\mathbf{C}$ are not. To illustrate that the non-identifiability may lead to quite severe confusion about inferences, consider a model with only two observed variables $\mathbf{x} = [x_1, x_2]^\top$ and $c_{11} = c_{22} = 1$. Two different hypothesis $\{b_{21}, c_{1L}, c_{2L}\} = \{0, 1, 1\}$ and $\{b_{21}, c_{1L}, c_{2L}\} = \{1, 1, -1\}$ with graphs shown in Figure 3 have equivalent factor models written as

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_L \end{bmatrix} \text{ and } \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} z'_1 \\ z'_2 \\ z'_L \end{bmatrix} .$$

The two models above have the same mixing matrix $\mathbf{D}$, up to permutation of columns $\mathbf{P}_\mathrm{f}$. In general we expect the number of solutions with equivalent distribution may be as large as 2^m, corresponding



(a)

(b)

Figure 3: Two DAGs with latent variables. They are equivalent if $\mathbf{z}$ has the same distribution as $\mathbf{z}'$.

to the number of times a column of $\mathbf{D}$ from its latent part (last m columns) con be exchanged with a column from its observed part (first d columns). This readily assumes that the sparsity pattern in $\mathbf{D}$ is identified, which follows from the results of Kagan et al. (1973).

One way to get identifiability is to change the distributions $\mathbf{z}_D$ and $\mathbf{z}_L$ such that they differ and cannot be exchanged. Here it is not enough to change the scale of the variables, that is, variance for continuous variables, because this effect can be countered by rescaling $\mathbf{C}$ with $\mathbf{S}_\mathrm{f}$. So we need distributions that differ beyond rescaling. In our examples we use Laplace and the more heavy-tailed Cauchy for $\mathbf{z}_D$ and $\mathbf{z}_L$, respectively. This specification is not unproblematic in practical situations however it can be sometimes restrictive and prone to model mismatch issues. We nevertheless show one practical example which leads to sensible inferences.

In time series applications for example, it is natural to go beyond an i.i.d. model for $\mathbf{z}$. One may for example use a Gaussian process prior for each factor to get smoothness over time, that is, $z_{j1},\ldots,z_{jN}|\nu_j \sim \mathcal{N}(0,\mathbf{K}_{\nu_j})$, where $\mathbf{K}_{\nu_j}$ is the covariance matrix with elements $k_{j,nn'} = k_{\upsilon_j,n}(n,n')$ and $k_{\upsilon_j,n}(\cdot)$ is the covariance function. For the i.i.d. Gaussian model the source distribution is only identifiable up to an arbitrary rotation matrix $\mathbf{U}$, that is, the rotated factors $\mathbf{U}\mathbf{z}$ are still i.i.d. . We can show that contrary to the i.i.d. Gaussian model, the Gaussian process factor model is identifiable if the covariance functions differ. We need to show that $\widehat{\mathbf{Z}} = \mathbf{U}\mathbf{Z}$ has a different covariance structure than $\mathbf{Z} = [\mathbf{z}_1 \ \ldots \ \mathbf{z}_N]$. We get $\overline{\mathbf{z}_n\mathbf{z}_{n'}^\top} = \mathrm{diag}(k_{1,nn'},\ldots,k_{d+m,nn'})$ and $\overline{\widehat{\mathbf{z}}_n\widehat{\mathbf{z}}_{n'}^\top} = \mathbf{U}\overline{\mathbf{z}_n\mathbf{z}_{n'}^\top}\mathbf{U}^\top = \mathbf{U}\mathrm{diag}(k_{1,nn'},\ldots,k_{d+m,nn'})\mathbf{U}^\top$ for the original and rotated variables, respectively. The covariances are indeed different and the model is thus identifiable if no covariance functions $k_{\upsilon_j,n}(n,n')$, $j = 1,\ldots,d+m$ are the same.

3. Prior Specification

In this section we provide a detailed description of the priors used for each one of the elements of our sparse linear identifiable model already defined in Equation (1). We start with $\boldsymbol{\varepsilon}$, the noise term that allow us to quantify the mismatch between a set of N observations $\mathbf{X} = [\mathbf{x}_1 \ \ldots \ \mathbf{x}_N]$ and the model itself. For this purpose, we use uncorrelated Gaussian noise components $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{\varepsilon}|\mathbf{0},\boldsymbol{\Psi})$ with conjugate inverse gamma priors for their variances as follows

$$\mathbf{X}|\mathbf{m},\boldsymbol{\Psi} \sim \prod_{n=1}^{N}\mathcal{N}(\mathbf{x}_n|\mathbf{m},\boldsymbol{\Psi}) \ ,$$

$$\boldsymbol{\Psi}^{-1}|s_s,s_r \sim \prod_{i=1}^{d}\mathrm{Gamma}(\psi_i^{-1}|s_s,s_r) \ ,$$

where we have already marginalized out $\boldsymbol{\varepsilon}$, $\boldsymbol{\Psi}$ is a diagonal covariance matrix denoting uncorrelated noise across dimensions and $\mathbf{m}$ is the mean vector such that $\mathbf{m}_{\mathrm{FM}} = \mathbf{C}\mathbf{z}_n$ and $\mathbf{m}_{\mathrm{DAG}} = \mathbf{B}\mathbf{x}_n + \mathbf{C}\mathbf{z}_n$. In the noise covariance hyperprior, s_s and s_r are the shape and rate, respectively. The selection of hyperparameters for $\mathbf{\Psi}$ should not be very critical as long as both "signal and noise" hypotheses are supported, that is, diffuse enough to allow for small values of ψ_i as well as for $\psi_i = 1$ (assuming that the data is standardized in advance). We set $s_s = 20$ and $s_r = 1$ in the experiments for instance. Another issue to consider when selecting s_s and s_r is the Bayesian analogue of the Heywood problem in which likelihood functions are bounded below away from zero as ψ_i tends to zero, hence inducing multi-modality in the posterior of ψ_i with one of the modes at zero. The latter can be avoided by specifying s_s and s_r such that the prior decays to zero at the origin, as we did above. It is well known, for example, that Heywood problems cannot be avoided using improper reference priors, $p(\psi_i) \propto 1/\psi_i$ (Martin and McDonald, 1975).

The remaining components of the model are described as it follows in five parts named sparsity, latent variables and driving signals, order search, allowing for correlated data and allowing for non-linearities. The first part addresses the interpretability of the model by means of parsimonious priors for $\mathbf{C}$ and $\mathbf{D}$. The second part describes the type of non-Gaussian distributions used on $\mathbf{z}$ in order to keep the model identifiable. The third part considers how a search over permutations of the observed variables can be used in order to handle the constraints imposed on matrix $\mathbf{R}$. The last two parts describe how introducing Gaussian process process priors in the model can be used to model non-independent observations and non-linear dependencies in the DAGs.

3.1 Sparsity

The use of sparse models will in many cases give interpretable results and is often motivated by the principle of parsimony. Also, in many application domains it is also natural from a prediction point of view to enforce sparsity because the number of explanatory variables may exceed the number of examples by orders of magnitude. In regularized maximum likelihood type formulations of learning (maximum a-posteriori) it has become popular to use one-norm (L_1) regularization for example to achieve sparsity (Tibshirani, 1996). In the fully Bayesian inference setting (with averaging over variables), the corresponding Laplace prior will not lead to sparsity because it is very unlikely for a posterior summary like the mean, median or mode to be estimated as exactly zero even asymptotically. The same effect can be expected from any continuous distribution used for sparsity like Student's t, α-stable and bimodal priors (continuous slab and spike priors, Ishwaran and Rao, 2005). Exact zeros can only be achieved by placing a point mass at zero, that is, explicitly specifying that the variable at hand is zero or not with some probability. This has motivated the introduction of many variants over the years of so-called slab and spike priors consisting of two component mixtures of a continuous part and a δ-function at zero (Lempers, 1971; Mitchell and Beauchamp, 1988; George and McCulloch, 1993; Geweke, 1996; West, 2003). In this paradigm, the columns of matrices $\mathbf{C}$ or $\mathbf{B}$ encode respectively, the connectivity of a factor or the set of parents associated to an observed variable. It is natural then to share information across elements in column j by assuming a common sparsity level $1 - \nu_j$, suggesting the following hierarchy

$$\begin{aligned} c_{ij}|q_{ij}, \cdot &\sim (1-q_{ij})\delta(c_{ij}) + q_{ij}\text{Cont}(c_{ij}|\cdot) \ , \\ q_{ij}|\nu_j &\sim \text{Bernoulli}(q_{ij}|\nu_j) \ , \\ \nu_j|\beta_m, \beta_p &\sim \text{Beta}(\nu_j|\beta_p\beta_m, \beta_p(1-\beta_m)) \ , \end{aligned} \tag{2}$$

where $\mathbf{Q}$, the binary matrix in Equation (1) appears naturally, $\delta(\cdot)$ is a Dirac δ-function, $\text{Cont}(\cdot)$ is the continuous slab component, $\text{Bernoulli}(\cdot)$ and $\text{Beta}(\cdot)$ are Bernoulli and beta distributions, respectively. Reparameterizing the beta distribution as $\text{Beta}(\nu_j|\alpha\beta/m, \beta)$ and taking the number of columns m of $\mathbf{Q} \odot \mathbf{C}$ to infinity, leads to the non-parametric version of the slab and spike model with a so-called Indian buffet process prior over the (infinite) masking matrix $\mathbf{Q} = \{q_{ij}\}$ (Ghahramani et al., 2006). Note also that $q_{ij}|\nu_j$ is mainly used for clarity to make the binary indicators explicit, nevertheless in practice we can work directly with $c_{ij}|\nu_j, \cdot \sim (1-\nu_j)\delta(c_{ij}) + \nu_j\mathrm{Cont}(c_{ij}|\cdot)$ because q_{ij} can be marginalized out.

As illustrated and pointed out by Lucas et al. (2006) and Carvalho et al. (2008) the model with a shared beta-distributed sparsity level per factor introduces the undesirable side-effect that there is strong co-variation between the elements in each column of the masking matrix. For example, in high dimensions we might expect that only a finite number of elements are non-zero, implying a prior favoring a very high sparsity rate $1-\nu_j$. Because of the co-variation, even the parameters that are clearly non-zero will have a posterior probability of being non-zero, $p(q_{ij} = 1|\mathbf{x}, \cdot)$, quite spread over the unit interval. Conversely, if our priors do not favor sparsity strongly, then the opposite situation will arise and the solution will become completely dense. In general, it is difficult to set the hyperparameters to achieve a sensible sparsity level. Ideally, we would like to have a model with a high sparsity level with high certainty about the non-zero parameters. We can achieve this by introducing a sparsity parameter η_{ij} for each element of $\mathbf{C}$ which has a mixture distribution with exactly this property

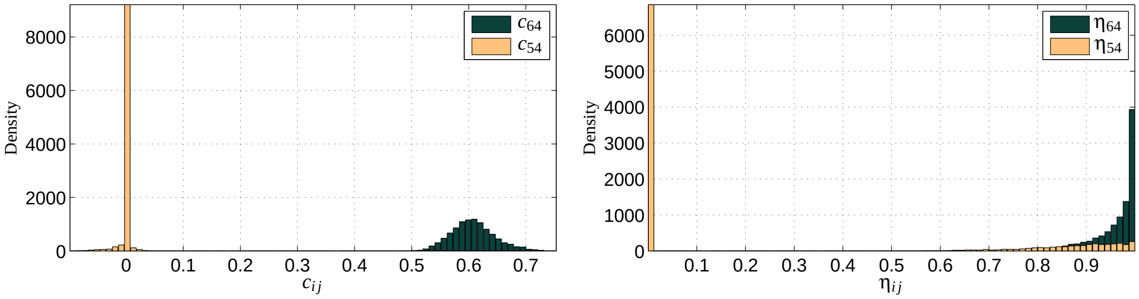
$$\begin{aligned} q_{ij}|\eta_{ij} &\sim \mathrm{Bernoulli}(q_{ij}|\eta_{ij}) \ , \\ \eta_{ij}|\nu_j, \alpha_p, \alpha_m &\sim (1-\nu_j)\delta(\eta_{ij}) + \nu_j \mathrm{Beta}(\eta_{ij}|\alpha_p\alpha_m, \alpha_p(1-\alpha_m)) \ . \end{aligned} \tag{3}$$

The distribution over η_{ij} expresses that we expect parsimony: either η_{ij} is zero exactly (implying that q_{ij} and c_{ij} are zero) or non-zero drawn from a beta distribution favoring high values, that is, q_{ij} and c_{ij} are non-zero with high probability. We use $\alpha_p = 10$ and $\alpha_m = 0.95$ which has mean $\alpha_m = 0.95$ and variance $\alpha_m(1-\alpha_m)/(1+\alpha_p) \approx 0.086$. The expected sparsity rate of the modified model is $(1-\alpha_m)(1-\nu_j)$. This model has the additional advantage that the posterior distribution of η_{ij} directly measures the distribution of $p(q_{ij} = 1|\mathbf{x}, \cdot)$. This is therefore the statistic for ranking/selection purposes. Besides, we may want to reject interactions with high uncertainty levels when the probability of $p(q_{ij} = 1|\mathbf{x}, \cdot)$ is less or very close to the expected value, $\alpha_m(1-\nu_j)$.

To complete the specification of the prior, we let the continuous slab part in Equation (2) be Gaussian distributed with inverse gamma prior on its variance. In addition, we scale the variances with ψ_i as

$$\begin{aligned} \mathrm{Cont}(c_{ij}|\psi_i, \tau_{ij}) &= \mathcal{N}(c_{ij}|0, \psi_i\tau_{ij}) \ , \\ \tau_{ij}^{-1}|t_s, t_r &\sim \mathrm{Gamma}(\tau_{ij}^{-1}|t_s, t_r) \ . \end{aligned} \tag{4}$$

This scaling makes the model easier to specify and tend to have better mixing properties (see Park and Casella, 2008). The slab and spike for $\mathbf{B}$ (DAG) is obtained from Equations (2), (3) and (4) by simply replacing c_{ij} with b_{ij} and q_{ij} with r_{ij}. As already mentioned, we use $\alpha_p = 10$ and $\alpha_m = 0.95$ for the hierarchy in Equation (3). For the column-shared parameter ν_j defined in Equation (2) we set the precision to $\beta_p = 100$ and consider the mean values for factor models and DAGs separately. For the factor model we set a diffuse prior by making $\beta_m = 0.9$ to reflect that some of the factors can be in general nearly dense or empty. For the DAG we consider two settings, if we expect to obtain dense graphs we set $\beta_m = 0.99$, otherwise we set $\beta_m = 0.1$. Both settings can produce sparse graphs, however smaller values of β_m increase the overall sparsity rate and the gap between $p(r_{ij} = 0)$ and $p(r_{ij} = 1)$. A large separation between these two probabilities makes interpretation easier and also



(a) (b)

Figure 4: Slab and spike prior example. (a) Posterior unnormalized densities for the magnitude of two particular elements of **C**. (b) Posterior density for $\eta_{ij} = p(c_{ij} \neq 0|\mathbf{x},\cdot)$. Here, $c_{64} \neq 0$ and $c_{54} = 0$ correspond to elements of the mixing matrix from the experiment shown in Figure 8.

helps to spot non-zeros (edges) with high uncertainty. The hyperparameters for the variance of the non-zero elements of **B** and **C** are set to get a diffuse prior distribution bounded away from zero ($t_s = 2$ and $t_r = 1$), to allow for a better separation between slab and spike components. For the particular case of $\mathbf{C}_L$, in principle the prior should not have support on zero at all, that is, the driving signal should not vanish, however for simplicity we allow this anyway as it has not given any problems in practice. Figure 4 shows a particular example of the posterior, $p(c_{ij},\eta_{ij}|\mathbf{x},\cdot)$ for two elements of **C** under the prior just described. In the example, $c_{64} \neq 0$ with high probability according to η_{ij}, whereas c_{54} is almost certainly zero since most of its probability mass is located exactly at zero, with some residual mass on the vicinity of zero, in Figure 4(a). In the one level hierarchy Equation (2) sparsity parameters are shared, $\eta_{64} = \eta_{54} = \nu_4$. The result would then be less parsimonious with the posterior density of ν_4 being spread in the unit interval with a single mode located close to β_m.

3.2 Latent Variables and Driving Signals

We consider two different non-Gaussian—heavy-tailed priors for **z**, in order to obtain identifiable factor models and DAGs. A wide class of continuous, unimodal and symmetric distributions in one dimension can be represented as infinite scale mixtures of Gaussians, which are very convenient for Gibbs-sampling-based inference. We focus on Student's t and Laplace distributions which have the following mixture representation (Andrews and Mallows, 1974)

$$\text{Laplace}(z|\mu,\lambda) = \int_0^\infty \mathcal{N}(z|\mu,\upsilon)\text{Exponential}(\upsilon|\lambda^2)d\upsilon\ , \tag{5}$$

$$t(z|\mu,\theta,\sigma^2) = \int_0^\infty \mathcal{N}(z|\mu,\upsilon\sigma^2)\text{Gamma}\left(\upsilon^{-1}\middle|\frac{\theta}{2},\frac{\theta}{2}\right)d\upsilon\ , \tag{6}$$

where $\lambda > 0$ is the rate, $\sigma^2 > 0$ the scale, $\theta > 0$ is the degrees of freedom, and the distributions have exponential and gamma mixing densities accordingly. For varying degrees of freedom θ, the t distribution can interpolate between very heavy-tailed (power law and Cauchy when $\theta = 1$) and very light tailed, that is, it becomes Gaussian when the degrees of freedom approaches infinity. The Laplace (or bi-exponential) distribution has tails which are intermediate between a t (with finite degrees of freedom) and a Gaussian. In this sense, the t distribution is more flexible but requires more careful selection of its hyperparameters because the model may become non-identifiable in the large θ limit (Gaussian).

An advantage of the Laplace distribution is that we can fix its parameter $\lambda = 1$ and let the model learn the appropriate scaling from $\mathbf{C}$ in Equation (1). If we use the pure DAG model, we will need to have a hyperprior for λ^2 in order to learn the variances of the latent variables/driving signals, as in Henao and Winther (2009). A hierarchical prior for the degrees of freedom in the t distribution is not easy to specify because there is no conjugate prior available with a standard closed form. Although a conjugate prior exists, is not straightforward to sample from it, since numerical integration must be used to compute its normalization constant. Another possibility is to treat θ as a discrete variable so computing the normalizing constant becomes straight forward.

Laplace and Student's t are not the only distributions admitting scale mixture representation. This mean that any other compatible type can be used as well, if the application requires it, and without considerable additional effort. Some examples include the logistic distribution (Andrews and Mallows, 1974), the stable family (West, 1987) and skewed versions of heavy-tailed distributions (Branco and Dey, 2001). Another natural extension to the mixtures scheme could be, for example, to set the mean of each component to arbitrary values and let the number of components be an infinite sum, thus ending up providing each factor with a Dirichlet process prior. This might be useful for cases when the latent factors are expected to be scattered in clusters due to the presence of subgroups in the data, as was shown by Carvalho et al. (2008).

3.3 Order Search

We need to infer the order of the variables in the DAG to meet the constraints imposed on $\mathbf{R}$ in Section 2. The most obvious way is to try to solve this task by inferring all parameters $\{\mathbf{P},\mathbf{B},\mathbf{C},\mathbf{z},\boldsymbol{\varepsilon}\}$ by a Markov chain Monte Carlo (MCMC) method such as Gibbs sampling. However, algorithms for searching over variable order prefer to work with models for which parameters other than $\mathbf{P}$ can be marginalized analytically (see Friedman and Koller, 2003; Teyssier and Koller, 2005). For our model, where we cannot marginalize analytically over $\mathbf{B}$ (due to $\mathbf{R}$ being binary), estimating $\mathbf{P}$ and $\mathbf{B}$ by Gibbs sampling would mean that we had to propose a new $\mathbf{P}$ for fixed $\mathbf{B}$. For example, exchanging the order of two variables would mean that they also exchange parameters in the DAG. Such a proposal would have very low acceptance, mainly as a consequence of the size of the search space and thus very poor mixing. In fact, for a given d number of variables there are $d!$ possible orderings $\mathbf{P}$, while there are $d!2^{(d(d+2m-1))/2}$ possible structures for $\{\mathbf{P},\mathbf{B},\mathbf{C}\}$. We therefore opt for an alternative strategy by exploiting the equivalence between factor models and DAGs shown in Section 2.1. In particular for $m = 0$, since $\mathbf{B}$ can be permuted to strictly lower triangular, then $\mathbf{D} = (\mathbf{I}-\mathbf{B})^{-1}\mathbf{C}_D$ can be permuted to triangular. This means that we can perform inference for the factor model to obtain $\mathbf{D}$ while searching in parallel for a set of permutations $\mathbf{P}$ that are in good agreement (in a probabilistic sense) with the triangular requirement of $\mathbf{D}$. Such a set of orderings is found during the inference procedure of the factor model. To set up the stochastic search, we need to modify the factor model slightly by introducing separate data (row) and factor (column) permutations, $\mathbf{P}$ and $\mathbf{P}_\mathrm{f}$ to obtain $\mathbf{x} = \mathbf{P}^\top\mathbf{D}\mathbf{P}_\mathrm{f}\mathbf{z}+\boldsymbol{\varepsilon}$. The reason for using two different permutation matrices, rather than only one like in the definition of the DAG model, is that we need to account for the permutation freedom of the factor model (see Section 2.1). Using the same permutation for row and column would thus require an additional step to identify the columns in the factor model. We make inference for the unrestricted factor model, but propose $\mathbf{P}^\star$ and $\mathbf{P}_\mathrm{f}^\star$ independently according to $q(\mathbf{P}^\star|\mathbf{P})q(\mathbf{P}_\mathrm{f}^\star|\mathbf{P}_\mathrm{f})$. Both distributions draw a new permutation matrix by exchanging two randomly chosen elements, for example, the order may change as $[x_1,x_2,x_3,x_4]^\top \to [x_1,x_4,x_3,x_2]^\top$. In other words, the proposals $q(\mathbf{P}^\star|\mathbf{P})$ and $q(\mathbf{P}_\mathrm{f}^\star|\mathbf{P}_\mathrm{f})$ are uniform distributions over the space of transpositions for $\mathbf{P}$ and $\mathbf{P}_\mathrm{f}$. Assuming we have no a-priori preferred ordering, we may use a Metropolis-Hastings (M-H) acceptance probability $\min(1,\xi_{\to\star})$ with $\xi_{\to\star}$ as a simple ratio of likelihoods with the permuted $\mathbf{D}$ masked to match the triangularity assumption. Formally, we use the binary mask $\mathbf{M}$ (containing zeros above the diagonal of its d first columns) and write

$$\xi_{\to\star} = \frac{\mathcal{N}(\mathbf{X}|(\mathbf{P}^\star)^\top(\mathbf{M}\odot\mathbf{P}^\star\mathbf{D}(\mathbf{P}_\mathrm{f}^\star)^\top)\mathbf{P}_\mathrm{f}^\star\mathbf{Z},\boldsymbol{\Psi})}{\mathcal{N}(\mathbf{X}|\mathbf{P}^\top(\mathbf{M}\odot\mathbf{P}\mathbf{D}\mathbf{P}_\mathrm{f}^\top)\mathbf{P}_\mathrm{f}\mathbf{Z},\boldsymbol{\Psi})}\ , \tag{7}$$

where $\mathbf{M}\odot\mathbf{D}$ is the masked $\mathbf{D}$ and $\mathbf{Z}=[\mathbf{z}_1\ \ldots\mathbf{z}_N]$. The procedure can be seen as a simple approach for generating hypotheses about good orderings, producing close to triangular versions of $\mathbf{D}$, in a model where the slab and spike prior provide the required bias towards sparsity. Once the inference is done, we end up having an estimate for the desired distribution over permutations $\mathbf{P}=\sum_i^{d!}\pi_i\delta_{\mathbf{P}_i}$, where $\boldsymbol{\pi}=[\pi_1\ \pi_2\ \ldots]$ is a sparse vector containing the probability for $\mathbf{P}=\mathbf{P}_i$, which in our case is proportional to the number of times permutation $\mathbf{P}_i$ was accepted by the M-H update during inference. Note that $\mathbf{P}_\mathrm{f}$ is just a nuisance variable that does not need to be stored or summarized.

3.4 Allowing for Correlated Data (CSLIM)

For the case where independence of observed variables cannot be assumed, for instance due to (time) correlation or smoothness, the priors discussed before for the latent variables and driving signals do not really apply anymore, however the only change we need to make is to allow elements in rows of $\mathbf{Z}$ to correlate. We can assume then independent Gaussian process (GP) priors for each latent variable instead of scale mixtures of Gaussians, to obtain what we have called correlated sparse linear identifiable modeling (CSLIM). For a set of N realizations of variable j we set

$$z_{j1},\ldots,z_{jN}|\upsilon_j\ \sim\ \mathrm{GP}(z_{j1},\ldots,z_{jN}|k_{\upsilon_j,n}(\cdot))\ , \tag{8}$$

where the covariance function has the form $k_{\upsilon_j,n}(n,n')=\exp(-\upsilon_j(n-n')^2)$, $\{n,n'\}$ is a pair of observation indices or time points and υ_j is the length scale controlling the overall level of correlation allowed for each variable (row) in $\mathbf{Z}$. Conceptually, Equation (8) implies that each latent variable j is sampled from a function and the GP acts as a prior over continuous functions. Since such a length scale is very difficult to set just by looking at the data, we further place priors on υ_j as

$$\upsilon_j|u_s,\kappa\ \sim\ \mathrm{Gamma}(\upsilon_j|u_s,\kappa)\ ,\quad \kappa|k_s,k_r\ \sim\ \mathrm{Gamma}(\kappa|k_s,k_r)\ . \tag{9}$$

Given that the conditional distribution of $\boldsymbol{\upsilon}=[\upsilon_1,\ldots,\upsilon_m]$ is not of any standard form, Metropolis-Hastings updates are used. In the experiments we use that $u_s=k_s=2$ and $k_r=0.02$. The details concerning inference for this model are given in Appendix A.

It is also possible to easily expand the possible applications of GP priors in this context by, for instance, using more structured covariance functions through scale mixture of Gaussian representations to obtain a prior distribution for continuous functions with heavy-tailed behavior—a t-processes (Yu et al., 2007), or learning the covariance function as well using inverse Wishart hyperpriors.

3.5 Allowing for Non-linearities (SNIM)

Provided that we know the true ordering of the variables, that is, $\mathbf{P}$ is known then $\mathbf{B}$ is surely strictly lower triangular. It is very easy to allow for non-linear interactions in the DAG model from Equation (1) by rewriting it as

$$\mathbf{P}\mathbf{x} = (\mathbf{R}\odot\mathbf{B})\mathbf{P}\mathbf{y} + (\mathbf{Q}\odot\mathbf{C})\mathbf{z} + \boldsymbol{\varepsilon} \,, \tag{10}$$

where $\mathbf{y} = [y_1,\ldots,y_d]^\top$ and $y_{i1},\ldots,y_{iN}|\upsilon_i \sim \mathrm{GP}(y_{i1},\ldots,y_{iN}|k_{\upsilon_i,x}(\cdot))$ has a Gaussian process prior with for instance, but not limited to, a stationary covariance function like $k_{\upsilon_i,x}(\mathbf{x},\mathbf{x}') = \exp(-\upsilon_i(\mathbf{x}-\mathbf{x}')^2)$, similar to Equation (8) and with the same hyperprior structure as in Equation (9). This is a straight forward extension that we call sparse non-linear multivariate modeling (SNIM) that is in spirit similar to Friedman and Nachman (2000), Hoyer et al. (2009), Zhang and Hyvärinen (2009), Zhang and Hyvärinen (2010) and Tillman et al. (2009), however instead of treating the inherent multiple regression problem in Equation (10) and the conditional independence of the observed variables independently, we proceed within our proposed framework by letting the multiple regressor be sparse, thus the conditional independences are encoded through $\mathbf{R}$. The main limitation of the model in Equation (10) is that if the true ordering of the variables is unknown, the exhaustive enumeration of $\mathbf{P}$ is needed. This means that this could be done for very small networks, for example, up to 5 or 6 variables. In principle, an ordering search procedure for the non-linear model only requires the latent variables $\mathbf{z}$ to have Gaussian process priors as well. The main difficulty is that in order to build covariance functions for $\mathbf{z}$ we need a set of observations that are not available because $\mathbf{z}$ is latent.

4. Model Comparison

Quantitative model comparison between factor models and DAGs is a key ingredient in SLIM. The joint probability of data $\mathbf{X}$ and parameters for the factor model part in Equation (1) is

$$p(\mathbf{X},\mathbf{C},\mathbf{Z},\boldsymbol{\varepsilon},\cdot) = p(\mathbf{X}|\mathbf{C},\mathbf{Z},\boldsymbol{\varepsilon})p(\mathbf{C}|\cdot)p(\mathbf{Z}|\cdot)p(\boldsymbol{\varepsilon})p(\cdot) \,,$$

where $(\cdot)$ indicates additional parameters in the hierarchical model. Formally the Bayesian model selection yardstick, the marginal likelihood for model $\mathcal{M}$

$$p(\mathbf{X}|\mathcal{M}) = \int p(\mathbf{X}|\boldsymbol{\Theta},\mathbf{Z})p(\boldsymbol{\Theta}|\mathcal{M})p(\mathbf{Z}|\mathcal{M})d\boldsymbol{\Theta}d\mathbf{Z} \,,$$

can be obtained by marginalizing the joint over the parameters $\boldsymbol{\Theta}$ and latent variables $\mathbf{Z}$. Computationally this is a difficult task because the marginal likelihood cannot be written as an average over the posterior distribution in a simple way. It is still possible using MCMC methods, for example by partitioning of the parameter space and multiple chains or thermodynamic integration (see Chib, 1995; Neal, 2001; Murray, 2007; Friel and Pettitt, 2008), but in general it must be considered as computationally expensive and non-trivial. On the other hand, evaluating the likelihood on a test set $\mathbf{X}^\star$, using predictive densities $p(\mathbf{X}^\star|\mathbf{X},\mathcal{M})$ is simpler from a computational point of view because it can be written in terms of an average over the posterior of the *intensive variables*, $p(\mathbf{C},\boldsymbol{\varepsilon},\cdot|\mathbf{X})$ and the prior distribution of the *extensive variables* associated with the test points,[2] $p(\mathbf{Z}^\star|\cdot)$ as

$$\mathcal{L}_{\mathrm{FM}} \overset{\mathrm{def}}{=} p(\mathbf{X}^\star|\mathbf{X},\mathcal{M}_{\mathrm{FM}}) = \int p(\mathbf{X}^\star|\mathbf{Z}^\star,\boldsymbol{\Theta}_{\mathrm{FM}},\cdot)p(\mathbf{Z}^\star|\cdot)p(\boldsymbol{\Theta}_{\mathrm{FM}},\cdot|\mathbf{X})d\mathbf{Z}^\star d\boldsymbol{\Theta}_{\mathrm{FM}}d(\cdot) \,, \tag{11}$$

[2] Intensive means not scaling with the sample size. Extensive means scaling with sample size in this case the size of the test sample.

where $\mathbf{\Theta}_{\rm FM} = \{\mathbf{C}, \boldsymbol{\varepsilon}\}$. This average can be approximated by a combination of standard sampling and exact marginalization using the scale mixture representation of the heavy-tailed distributions presented in Section 3.2. For the full DAG model in Equation (1), we will not average over permutations $\mathbf{P}$ but rather calculate the test likelihood for a number of candidates $\mathbf{P}^{(1)}, \ldots, \mathbf{P}^{(c)}, \ldots$ as

$$\begin{aligned} \mathcal{L}_{\rm DAG} &\overset{\rm def}{=} p(\mathbf{X}^\star|\mathbf{P}^{(c)}, \mathbf{X}, \mathcal{M}_{\rm DAG}) \ , \\ &= \int p(\mathbf{X}^\star|\mathbf{P}^{(c)}, \mathbf{X}, \mathbf{Z}^\star, \mathbf{\Theta}_{\rm DAG}, \cdot) p(\mathbf{Z}^\star|\cdot) p(\mathbf{\Theta}_{\rm DAG}, \cdot|\mathbf{X}) d\mathbf{Z}^\star d\mathbf{\Theta}_{\rm DAG} d(\cdot) \ , \end{aligned} \tag{12}$$

where $\mathbf{\Theta}_{\rm DAG} = \{\mathbf{B}, \mathbf{C}, \boldsymbol{\varepsilon}\}$. We use sampling to compute the test likelihoods in Equations (11) and (12). With Gibbs, we draw samples from the posterior distributions $p(\mathbf{\Theta}_{\rm FM}, \cdot|\mathbf{X})$ and $p(\mathbf{\Theta}_{\rm DAG}, \cdot|\mathbf{X})$, where $(\cdot)$ is shorthand for example for the degrees of freedom θ, if Student t distributions are used. The average over the extensive variables associated with the test points $p(\mathbf{Z}^\star|\cdot)$ is slightly more complicated because naively drawing samples from $p(\mathbf{Z}^\star|\cdot)$ results in an estimator with high variance—for $\psi_i \ll \upsilon_{jn}$. Instead we exploit the infinite mixture representation to marginalize exactly $\mathbf{Z}^\star$ and then draw samples in turn for the scale parameters. Omitting the permutation matrices for clarity, in general we get

$$\begin{aligned} p(\mathbf{X}^\star|\mathbf{\Theta}, \cdot) &= \int p(\mathbf{X}^\star|\mathbf{Z}^\star, \mathbf{\Theta}, \cdot) p(\mathbf{Z}^\star|\cdot) d\mathbf{Z}^\star \ , \\ &= \prod_n \int \mathcal{N}(\mathbf{x}_n^\star|\mathbf{m}_n, \mathbf{\Sigma}_n) \prod_j p(\upsilon_{jn}|\cdot) d\upsilon_{jn} \approx \frac{1}{N_{\rm rep}} \prod_n \sum_r^{N_{\rm rep}} \mathcal{N}(\mathbf{x}_n^\star|\mathbf{m}_n, \mathbf{\Sigma}_n) \ , \end{aligned}$$

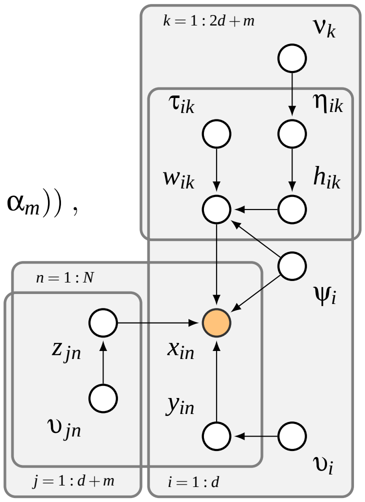
where $N_{\rm rep}$ is the number of samples generated to approximate the intractable integral ($N_{\rm rep} = 500$ in the experiments). For the factor model $\mathbf{m}_n = \mathbf{0}$ and $\mathbf{\Sigma}_n = \mathbf{C}_D \mathbf{U}_n \mathbf{C}_D^\top + \mathbf{\Psi}$. For the DAG, $\mathbf{m}_n = \mathbf{B}\mathbf{x}_n^\star$ and $\mathbf{\Sigma}_n = \mathbf{C}\mathbf{U}_n\mathbf{C}^\top + \mathbf{\Psi}$. The covariance matrix $\mathbf{U}_n = \mathrm{diag}(\upsilon_{1n}, \ldots, \upsilon_{(d+m)n})$ with elements υ_{jn}, is sampled directly from the prior, accordingly. Once we have computed $p(\mathbf{X}^\star|\mathbf{\Theta}_{\rm FM}, \cdot)$ for the factor model and $p(\mathbf{X}^\star|\mathbf{\Theta}_{\rm DAG}, \cdot)$ for the DAG, we can use them to average over $p(\mathbf{\Theta}_{\rm FM}, \cdot|\mathbf{X},)$ and $p(\mathbf{\Theta}_{\rm DAG}, \cdot|\mathbf{X})$ to obtain the predictive densities $p(\mathbf{X}^\star|\mathbf{X}, \mathcal{M}_{\rm FM})$ and $p(\mathbf{X}^\star|\mathbf{X}, \mathcal{M}_{\rm DAG})$, respectively.

For the particular case in which $\mathbf{X}$ and consequently $\mathbf{Z}$ are correlated variables—CSLIM, we use a slightly different procedure for model comparison. Instead of using a test set, we randomly remove some proportion of the elements of $\mathbf{X}$ and perform inference with missing values, then we summarize the likelihood on the missing values. In particular, for the factor model we use $\mathbf{M}_{\rm miss} \odot \mathbf{X} = \mathbf{M}_{\rm miss} \odot (\mathbf{Q}_L \odot \mathbf{C}_L \mathbf{Z} + \boldsymbol{\varepsilon})$ where $\mathbf{M}_{\rm miss}$ is a binary masking matrix with zeros corresponding to test points, that is, the missing values. See details in Appendix A. Note that this scheme is not exclusive to CSLIM thus can be also used with SLIM or when the observed data contain actual missing values.

5. Model Overview and Practical Details

The three models described in the previous section namely SLIM, CSLIM and SNIM can be summarized as a graphical model and as a probabilistic hierarchy as follows

$$
\begin{aligned}
\mathbf{x}_n|\mathbf{W},\mathbf{y}_n,\mathbf{z}_n,\boldsymbol{\Psi} &\sim \mathcal{N}(\mathbf{x}_n|\mathbf{W}[\mathbf{y}_n\ \mathbf{z}_n]^\top,\boldsymbol{\Psi})\ ,\ \ \mathbf{W}=[\mathbf{B}\ \mathbf{C}]\ ,\\
\psi_i^{-1}|s_s,s_r &\sim \mathrm{Gamma}(\psi_i^{-1}|s_s,s_r)\ ,\\
w_{ik}|h_{ik},\psi_i,\tau_{ik} &\sim (1-h_{ik})\delta_0(w_{ik})+h_{ik}\mathcal{N}(w_{ik}|0,\psi_i\tau_{ik})\ ,\\
h_{ik}|\eta_{ik} &\sim \mathrm{Bernoulli}(h_{ik}|\eta_{ik})\ ,\ \ \mathbf{H}=[\mathbf{R}\ \mathbf{Q}]\ ,\\
\eta_{ik}|\nu_k,\alpha_p,\alpha_m &\sim (1-\nu_k)\delta(\eta_{ik})+\nu_k\mathrm{Beta}(\eta_{ik}|\alpha_p\alpha_m,\alpha_p(1-\alpha_m))\ ,\\
\nu_k|\beta_m,\beta_p &\sim \mathrm{Beta}(\nu_k|\beta_p\beta_m,\beta_p(1-\beta_m))\ ,\\
\tau_{ik}^{-1}|t_s,t_r &\sim \mathrm{Gamma}(\tau_{ik}^{-1}|t_s,t_r)\ ,\\
z_{j1},\dots,z_{jN}|\upsilon &\sim \begin{cases}\prod_n \mathcal{N}(z_{jn}|0,\upsilon_{jn})\ , & \text{(SLIM)}\\ \mathrm{GP}(z_{j1},\dots,z_{jN}|k_{\upsilon_j,n}(\cdot))\ , & \text{(CSLIM)}\end{cases}\\
y_{i1},\dots,y_{iN}|\upsilon &\sim \begin{cases}x_{i1},\dots,x_{iN}\ , & \text{(SLIM)}\\ \mathrm{GP}(y_{i1},\dots,y_{iN}|k_{\upsilon_i,x}(\cdot))\ , & \text{(SNIM)}\end{cases}
\end{aligned}
$$



where we have omitted **P** and the hyperparameters in the graphical model. Latent variable and driving signal parameters υ can have one of several priors: Exponential$(\upsilon|\lambda^2)$ (Laplace), Gamma$(\upsilon^{-1}|\theta/2,\theta/2)$ (Student's t) or Gamma$(\upsilon|u_s,\kappa)$ (GP), see Equations (5), (6) and (9), respectively. The latent variables/driving signals z_{jn} and the mixing/connectivity matrices with elements c_{ij} or b_{ij} are modeled independently. Each element in **B** and **C** has its own slab variance τ_{ij} and probability of being non-zero η_{ij}. Moreover, there is a shared sparsity rate per column ν_k. Variables υ_{jn} are variances if z_{jn} use a scale mixture of Gaussian's representation, or length scales in the GP prior case. Since we assume no sparsity for the driving signals, $\eta_{ik}=1$ for $d+i=k$ and $\eta_{ik}=0$ for $d+i\neq k$. In addition, we can recover the pure DAG by making $m=0$ and the standard factor model by making instead $\eta_{ik}=0$ for $k\leq 2d$. All the details for the Gibbs sampling based inference are summarized in appendix A.

5.1 Proposed Workflow

We propose the workflow shown in Figure 1 to integrate all elements of SLIM, namely factor model and DAG inference, stochastic order search and model selection using predictive densities.

1. Partition the data into $\{\mathbf{X},\mathbf{X}^\star\}$.

2. Perform inference on the factor model and stochastic order search. One Gibbs sampling update consists of computing the conditional posteriors in Equations (13), (14), (15), (16), (17), (18) and (19) in sequence, followed by several repetitions (we use 10) of the M-H update in Equation (7) for the permutation matrices **P** and $\mathbf{P}_\mathrm{f}$.

3. Summarize the factor model, mainly **C**, $\{\eta_{ij}\}$ and $\mathcal{L}_{\mathrm{FM}}$ using quantiles (0.025, 0.5 and 0.975).

4. Summarize the orderings, **P**. Select the top m_{top} candidates according to their frequency during inference in step 2.

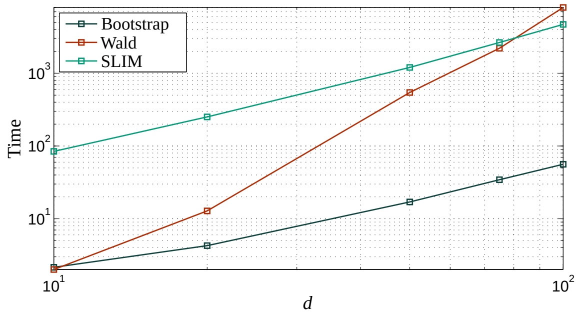


Figure 5: Runtime comparison.

5. Perform inference on the DAGs for each one of the ordering candidates, $\mathbf{P}^{(1)}, \ldots, \mathbf{P}^{(m_\text{top})}$ using Gibbs sampling by computing Equations (13), (14), (15), (16), (17), (18) and (19) in sequence, up to minor changes described in Appendix A.

6. Summarize the DAGs, $\mathbf{B}$, $\mathbf{C}_L$, $\{\eta_{ik}\}$ and $\mathcal{L}_\text{DAG}^{(1)}, \ldots, \mathcal{L}_\text{DAG}^{(m_\text{top})}$ using quantiles (0.025, 0.5 and 0.975). Note that $\{\eta_{ik}\}$ contains non-zero probabilities for $\mathbf{R}$ and $\mathbf{Q}$ corresponding to $\mathbf{B}$ and $\mathbf{C}_L$, respectively.

We use medians to summarize all quantities in our model because $\mathbf{D}$, $\mathbf{B}$ and $\{\eta_{ik}\}$ are bimodal while the remaining variables are in general skewed posterior distributions. Inference with GP priors for time series data (CSLIM) or non-linear DAGs (SNIM) is fairly similar to the i.i.d. case, see Appendix A for details. Source code for SLIM and all its variants proposed so far has been made available at `http://cogsys.imm.dtu.dk/slim/` as Matlab scripts.

5.2 Computational Cost

The cost of running the linear DAG with latent variables or the factor model is roughly the same, that is, $O(N_s d^2 N)$ where N_s is the total number of samples including the burn-in period. The memory requirements on the other hand are approximately $O(N_p d^2)$ if all the samples after the burn-in period N_p are stored. This means that the inference procedures scale reasonably well if N_s is kept in the lower ten thousands. The non-linear version of the DAG is considerably more expensive due to the GP priors, hence the computational cost rises up to $O(N_s(d-1)N^3)$.

The computational cost of LiNGAM, being the closest to our linear models, is mainly dependent on the statistic used to prune/select the model. Using bootstrapping results in $O(N_b^3)$, where N_b is the number of bootstrap samples. The Wald statistic leads to $O(d^6)$, while Wald with χ^2 second order model fit test amounts to $O(d^7)$. As for the memory requirements, bootstrapping is very economic whereas Wald-based statistics require $O(d^6)$.

The method for non-linear DAGs described in Hoyer et al. (2009) is defined for a pair of variables, and it uses GP-based regression and kernelized independence tests. The computational cost is $O(N_g N^3)$ where N_g is the number of gradient iterations used to maximize the marginal likelihood of the GP. This is the same order of complexity as our non-linear DAG sampler.

Figure 5 shows average running times in a standard desktop machine (two cores, 2.6GHz and 4Gb RAM) over 10 different models with $N = 1000$ and $d = \{10, 20, 50, 100\}$. As expected, LiNGAM with bootstrap is very fast compared to the others while our model approaches LiNGAM with Wald statistic as the number of observations increases. We did not include LiNGAM with second order model fit because for $d = 50$ it is already prohibitive. For this small test we used a C implementation of our model with $N_s = 19000$. We are aware that the performance of a C and a Matlab implementation can be different, however we still do the comparison because the most expensive operations in the Matlab code for LiNGAM are computed through BLAS routines not involving large loops, thus a C implementation of LiNGAM should not be noticeably faster than its Matlab counterpart.

6. Simulation Results

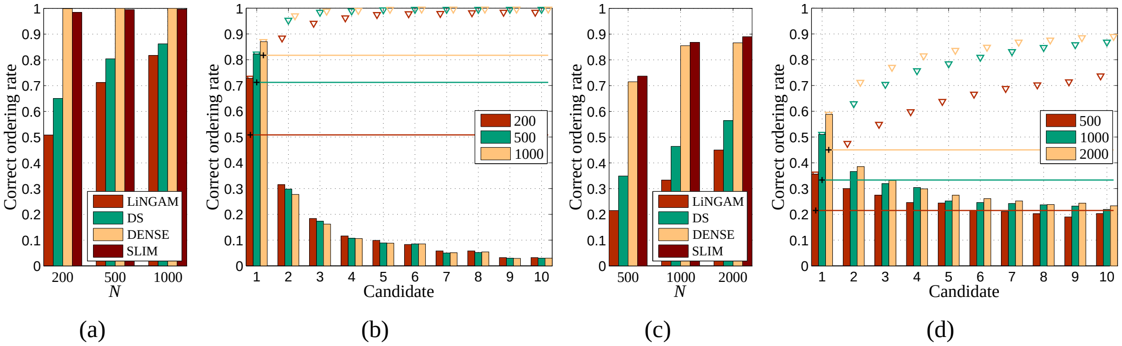
We consider six sets of experiments to illustrate the features of SLIM. In our comparison with other methods we focus on the DAG structure learning part because it is somewhat easier to benchmark a DAG than a factor model. However, we should stress that DAG learning is just one component of SLIM. Both types of model and their comparison are important, as will be illustrated through the experiments. For the reanalysis of flow cytometry data using our models, quantitative model comparison favors the DAG with latent variables rather than the standard factor model or the pure DAG which was the paradigm used in the structure learning approach of Sachs et al. (2005).

The first two experiments consist of extensive tests using artificial data in a setup originally from LiNGAM and network structures taken from the Bayesian net repository. We test the features of SLIM and compare with LiNGAM and some other methods in settings where they have proved to work well. The third set of experiments addresses model comparison, the fourth and fifth present results for our DAG with latent variables and the non-linear DAG (SNIM) on both artificial and real data. The sixth uses real data previously published by Sachs et al. (2005) and the last one provides simple results for a factor model using Gaussian process priors for temporal smoothness (CSLIM), tested on a time series gene expression data set (Kao et al., 2004). In all cases we ran 10000 samples after a burn-in period of 5000 for the factor model, and a single chain with 3000 samples and 1000 as burn-in iterations for the DAG, that is, $N_s = 19000$ used in the computational cost comparison. As a summary statistic we use median values everywhere, and Laplace distributions for the latent factors if not stated otherwise.

6.1 Artificial Data

We evaluate the performance of our model against LiNGAM,[3] using the artificial model generator presented and fully explained in Shimizu et al. (2006). Concisely, the generator produces both dense and sparse networks with different degrees of sparsity, $\mathbf{Z}$ is generated from a heavy-tailed non-Gaussian distribution through a generalized Gaussian distribution with zero mean, unit variance and random shape, $\mathbf{X}$ is generated recursively using Equation (1) with $m = 0$ and then randomly permuted to hide the correct order, $\mathbf{P}$. Approximately, half of the networks are fully connected while the remaining portion comprises sparsity levels between 10% and 80%. Having dense networks (0% sparsity) in the benchmark is crucial because in such cases the correct order of the variables is unique, thus more difficult to find. This setup is particularly challenging because the model needs to identify both dense and sparse models. For the experiment we have generated 1000 different data set/models using $d = \{5, 10\}$, $N = \{200, 500, 1000, 2000\}$ and the DAG was selected using the median of the training likelihood, $p(\mathbf{X}|\mathbf{P}_\text{r}^{(k)}, \mathbf{R}^{(k)}, \mathbf{B}^{(k)}, \mathbf{C}_D^{(k)}, \mathbf{Z}, \mathbf{\Psi}, \cdot)$, for $k = 1, \ldots, m_\text{top}$.

[3] Matlab package (v.1.42) available at `http://www.cs.helsinki.fi/group/neuroinf/lingam/`.

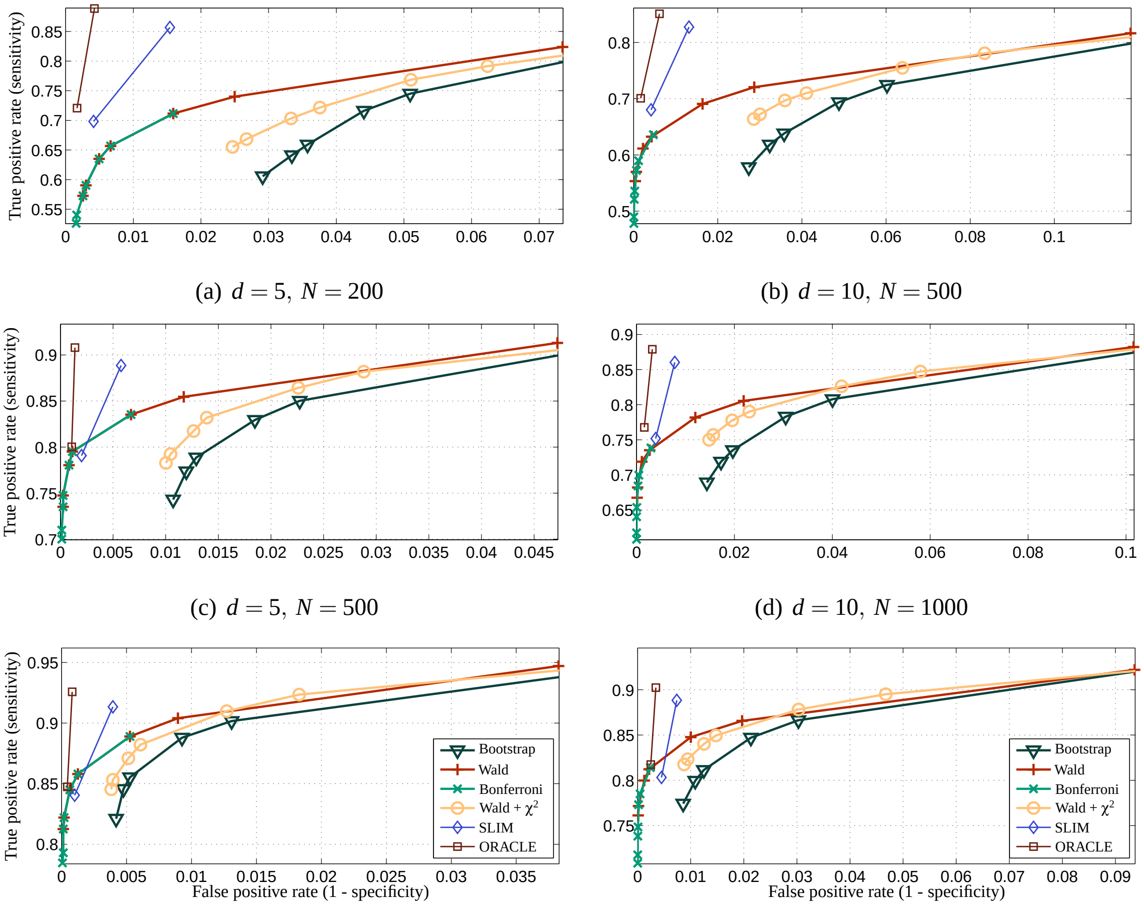


(a) (b) (c) (d)

Figure 6: Ordering accuracies for LiNGAM suite using $d = 5$ in (a,b) and $d = 10$ in (c,d). (a,c) Total correct ordering rates where DENSE is our factor model without sparsity prior and DS corresponds to DENSE but using the deterministic ordering search used in LiNGAM. (b,c) Correct ordering rate vs. candidates from SLIM. The crosses and horizontal lines correspond to LiNGAM while the triangles are accumulated correct orderings across candidates used by SLIM.

6.1.1 ORDER SEARCH

With this experiment we want to quantify the impact of using sparsity, stochastic ordering search and more than one ordering candidate, that is, $m_{\text{top}} = 10$ in total. Figure 6 evaluates the proportion of correct orderings for different settings. We have the following abbreviations for this experiment, DENSE is our factor model without sparsity prior, that is, assuming that $p(r_{ij} = 1) = 1$ a priori. DS (deterministic search) assumes no sparsity as in DENSE but replaces our stochastic search for permutations with the deterministic approach used by LiNGAM, that is, we replace the M-H update from Equation (7) by the procedure described next: after inference we compute $\mathbf{D}^{-1}$ followed by a column permutation search using the Hungarian algorithm and a row permutation search by iterative pruning until getting a version of $\mathbf{D}$ as triangular as possible (Shimizu et al., 2006). Several comments can be made from the results, (i) For $d = 5$ there is no significant gain for increasing N, mainly because the size of the permutation space is small, that is, 5!. (ii) The difference in performance between SLIM and DENSE is not significative because we look for triangular matrices in a probabilistic sense, hence there is no real need for exact zeros but just very small values, this does not mean that the sparsity in the factor model is unnecessary, on the contrary we still need it if we want to have readily interpretable mixing matrices. (iii) Using more than one ordering candidate considerably improves the total correct ordering rate, for example, by almost 30% for $d = 5$, $N = 200$ and 35% for $d = 10$, $N = 500$. (iv) The number of accumulated correct orderings found saturates as the number of candidates used increases, suggesting that further increasing m_{top} will not considerably change the overall results. (v) The number of correct orderings tends to accumulate on the first candidate when N increases since the uncertainty of the estimation of the parameters in the factor model decreases accordingly. (vi) When the network is not dense, it could happen that more than one candidate has a correct ordering, hence the total rates (triangles) are not just the sum of the bar heights in Figures 6(b) and 6(d). (vii) It seems that except for $d = 10$, $N = 5000$ it is enough to consider just the first candidate in SLIM to obtain as many correct orderings as LiNGAM does. (viii) From Figures 6(a) and 6(c), the three variants of SLIM considered perform better than LiNGAM,



(a) $d = 5,\ N = 200$

(b) $d = 10,\ N = 500$

(c) $d = 5,\ N = 500$

(d) $d = 10,\ N = 1000$

(e) $d = 5,\ N = 1000$

(f) $d = 10,\ N = 2000$

Figure 7: Performance measures for LiNGAM suite. Results include the settings: $d = \{5,10\}$, $N = \{200,500,1000,2000\}$, four model selectors for LiNGAM (bootstrap, Wald, Bonferroni and Wald + χ^2 statistics) and seven p-value cutoffs for the statistics used in LiNGAM (0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5). ORACLE corresponds to oracle results for SLIM, both computed for two settings: diffuse $\beta_m = 0.99$ and sparse $\beta_m = 0.1$ priors. Markers close to the top-left corner denote better results in average.

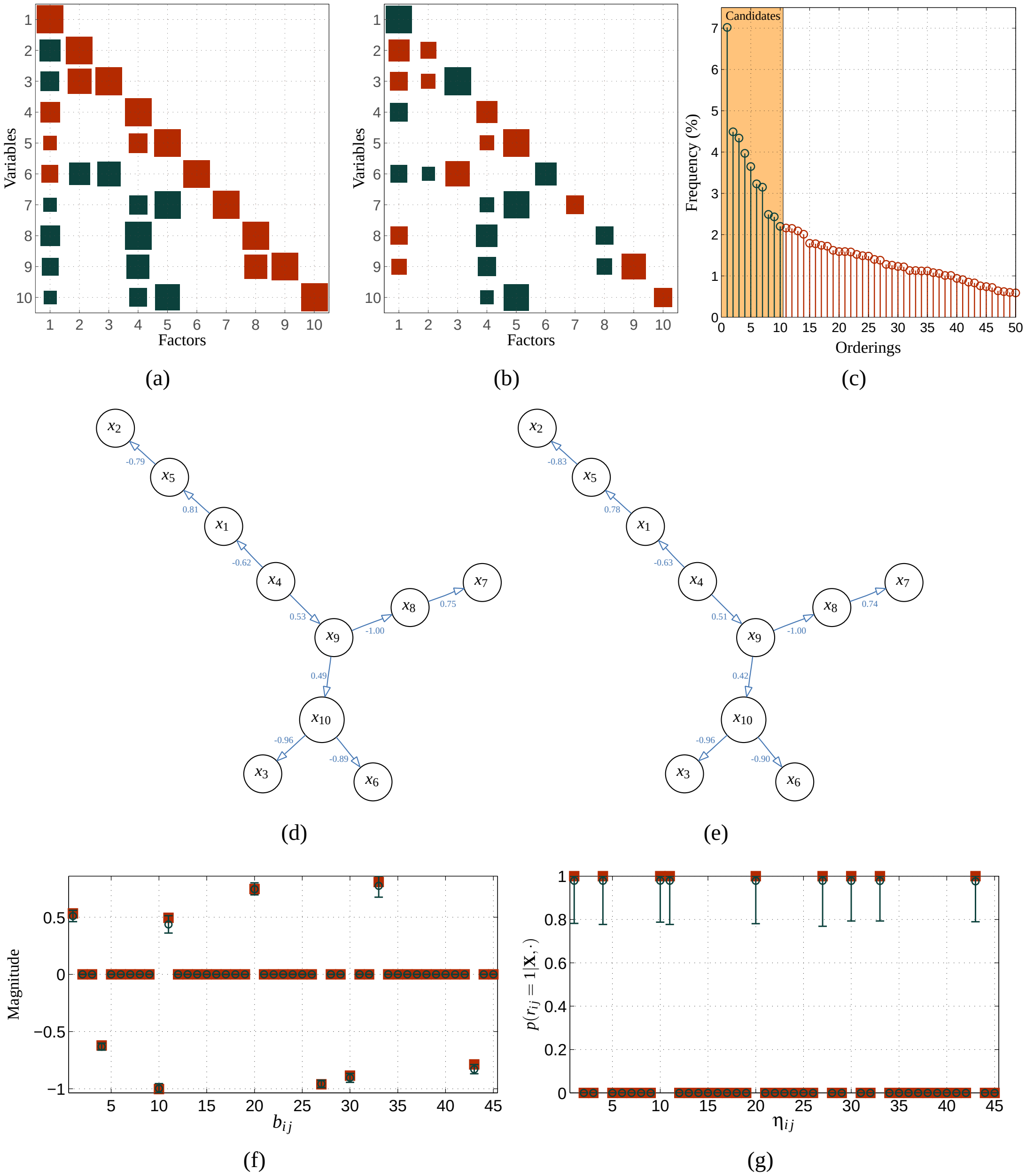
even when using the same single candidate ordering search proposed by Shimizu et al. (2006). (ix) In some cases the difference between SLIM and LiNGAM is very large, for example, for $d = 10$ using two candidates and $N = 1000$ is enough to obtain as many correct orderings as LiNGAM with $N = 5000$.

6.1.2 DAG LEARNING

Now we evaluate the ability of our model to capture the DAG structure in the data, provided the permutation matrices obtained in the previous stage as a result of our stochastic order search. Results are summarized in Figure 7 using receiving operating characteristic (ROC) curves. The true and false positive rates are averaged over the number of trials (1000) for each setting to make the scaling in the plots more meaningful given the various levels of sparsity considered. The rates are computed in the usual way, however it must be noted that the true number of absent links in a network can be as large as $d(d-1)$, that is, twice the number of links in a DAG, because in the case of an estimated DAG based in a wrong ordering the number of false positives can sum up to $d(d-1)/2$ even if the true network is not empty. For LiNGAM we use four different statistics to prune the DAG after the ordering has been found, namely bootstrapping, Wald, Bonferroni and Wald with second order χ^2 model fit test. In every case we run LiNGAM for 7 different p-value cutoffs, namely, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1 and 0.5 to build the ROC curve. For SLIM we consider the two settings for β_m discussed in Section 3.1, that is, a diffuse prior supporting the existence of dense graphs, $\beta_m = 0.99$ and $\beta_m = 0.1$. In order to test how good SLIM is at selecting one DAG out of the m_{top} candidates, we also report the oracle results under the name of ORACLE, where in every case we select the candidate with less error instead of $\text{argmax}_k \; p(\mathbf{X}|\mathbf{P}_{\text{r}}^{(k)},\mathbf{R}^{(k)},\mathbf{B}^{(k)},\mathbf{C}_D^{(k)},\mathbf{Z},\mathbf{\Psi},\cdot)$. Using $\beta_m = 0.99$ is not very useful in practice because in a real situation we expect that the underlying DAG is sparse, however the LiNGAM suite has as many dense graphs as sparse ones making $\beta_m = 0.1$ a poor choice. From Figure 7, it is clear that for $\beta_m = 0.99$, SLIM is clearly superior, providing the best true positive rate (TPR) - false positive rate (FPR) tradeoff. For $\beta_m = 0.1$ there is no real difference between SLIM and some settings of LiNGAM (Wald and Bonferroni). Concerning SLIM's model selection procedure, it can be seen that the difference between SLIM and ORACLE nicely decreases as the number of observations increases. We also tested the DAG learning procedure in SLIM when the true ordering is known (results not shown) and we found only a very small difference compared to ORACLE. It is important to mention that further increasing or reducing β_m does not significantly change the results shown; this is because β_m does not fully control the sparsity of the model, thus even for $\beta_m = 1$ the model will be still sparse due to element-wise link confidence, α_m. As for LiNGAM, it seems that Wald performs better than Wald + χ^2, however just by looking at Figure 7, it is to be expected that for larger N the latter perform better because the Wald statistic alone will tend to select more dense models.

6.1.3 ILLUSTRATIVE EXAMPLE

Finally we want to show some of the most important elements of SLIM taking one successfully estimated example from the LiNGAM suite. Figure 8 shows results for a particular DAG with 10 variables obtained using 500 observations, see Figures 8(d) and 8(e) for the ground truth and the estimated DAG, respectively. True and estimated mixing matrices $\mathbf{D}$ for the equivalent factor model are also shown in Figures 8(a) and 8(b), respectively. In total our algorithm produced 92 orderings out of 3.6×10^6 possible, from which all $m_{\text{top}} = 10$ candidates were correct. Figure 8(c) shows the first 50 candidates and their frequency during sampling, the shaded area encloses the $m_{\text{top}} = 10$ candidates. From Figure 8(f) we see that the elements of $\mathbf{B}$ are correctly estimated and their credible intervals are small, mainly due to the lack of model mismatch. Figure 8(g) shows a good separation between zero and non-zero elements of $\mathbf{B}$ as summarized by $p(r_{ij} = 1|\mathbf{X},\cdot)$. It is worthwhile mentioning that using $\beta_m = 0.99$ instead of $\beta_m = 0.1$ in this example, still produces the right DAG, although the separation between zero and non-zero elements in Figure 8(g) will be smaller and with higher uncertainty, that is, larger credible intervals.



(a)

(b)

(c)

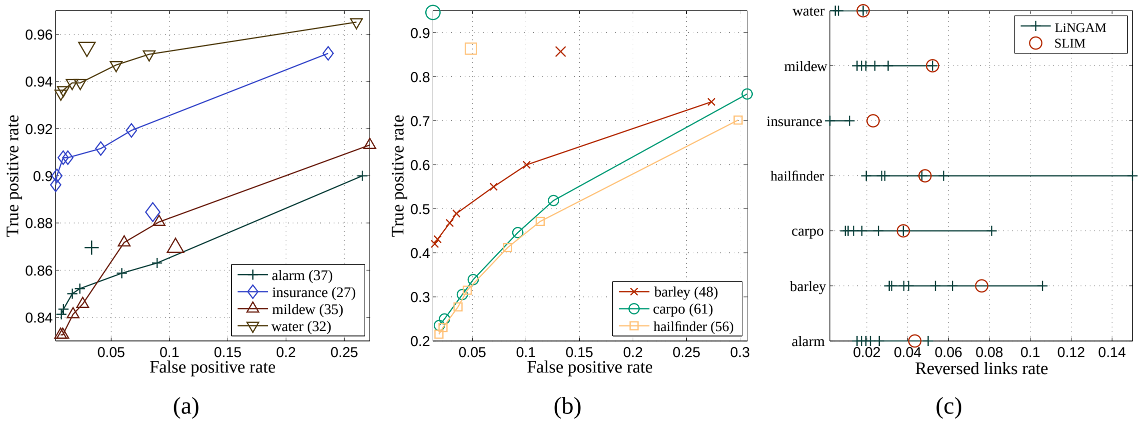
(d)

(e)

(f)

(g)

Figure 8: Ground truth and estimated structures. (a) Ground truth mixing matrix. (b) Estimated mixing matrix using our sparse factor model. Note the sign ambiguity in some of the columns. (c) First 50 (out of 92) ordering candidates produced by our method during inference and their frequency, the first m_{top} candidates were used for to learn DAGs. (d) Ground truth DAG. (e) Top candidate estimated using SLIM. (f) Estimated median weights for the DAG including 95% credible intervals and ground truth (squares). (g) Summary of link probabilities measured as $\eta_{ij} = p(r_{ij} = 1|\mathbf{X}, \cdot)$.



(a) (b) (c)

Figure 9: Performance measures for the Bayesian networks repository experiments. Each connected marker correspond to a different p-value in LiNGAM, starting left to right from 0.005. Disconnected markers denote SLIM results. Numbers in parentheses indicate number of variables.

6.2 Bayesian Networks Repository

Next we want to compare our method against LiNGAM on some realistic structures. We consider 7 well known benchmark structures from the Bayesian network repository,[4] namely alarm, barley, carpo, hailfinder, insurance, mildew and water ($d = 37, 48, 61, 56, 27, 35, 32$ respectively). Since we do not have continuous data for any of the structures, we generated 10 data sets of size $N = 500$ for each of them using heavy-tailed distributions with different parameters and Equation (1) with $m = 0$, in a similar way as we did for the previous set of experiments, with $\mathbf{R}$ set to the ground truth and $\mathbf{B}$ from $\text{sign}(\mathcal{N}(0,1)) + \mathcal{N}(0,0.2)$. For LiNGAM, we only use Wald statistics because as seen in the previous experiment, it performs significantly better that bootstrapping. Again, we estimate models for different p-value cutoffs (0.0005, 0.001, 0.005, 0.01, 0.05, 0.1 and 0.5). For SLIM, we set $\beta_m = 0.1$ since all the networks in the repository are sparse. Figures 9(a), 9(b) and 9(c) show averaged performance measures respectively as ROC curves and the proportion of links reversed in the estimated model due to ordering errors.

In this case, the results are mixed when looking at the performances obtained. Figure 9(b) shows that SLIM is better than LiNGAM in the larger data sets with a significant difference. Figure 9(a) shows for the remaining four data sets, that LiNGAM is better in two cases corresponding to the insurance and mildew networks. In general, both methods perform reasonably well given the size of the problems and the amount of data used to fit the models. However, SLIM tends to be more stable, when looking at the range of the true positive rates. It is important to note that the best and worst case for SLIM correspond to the largest and smallest network, respectively. We do not have a sensible explanation about why SLIM is performing that poorly on the insurance network. Figure 9(c) implicitly reveals that both methods are unable to find the right ordering of the variables.

We also tried the following methods with encoded Gaussian assumptions: standard DAG search, order search, sparse candidate pruning then DAG search (Friedman et al., 1999), L1MB then DAG

[4] Network structures available at `http://compbio.cs.huji.ac.il/Repository/`.

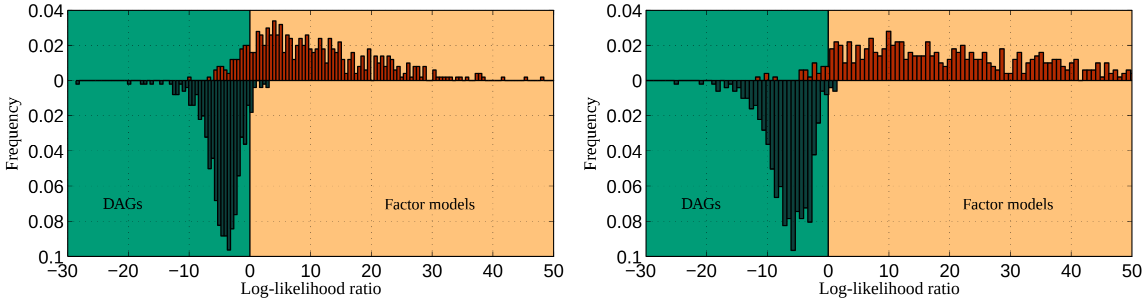
search (Schmidt et al., 2007), and sparse candidate pruning then order search (Teyssier and Koller, 2005). We observed (results not shown) that these methods produce similar results to those obtained by either LiNGAM or SLIM when only looking at the resulting undirected graph, that is, removing the directionality of the links. Evaluation of directionality in Gaussian models is out of the question because such methods can only find DAGs up to Markov equivalence classes, thus evaluation must be made using partially directed acyclic graphs (PDAGs). It is still possible to modify some of the methods mentioned above to handle non-Gaussian data by for instance using some other appropriate conditional independence tests, however this is out of the scope of this paper.

6.3 Model Comparison

In this experiment we want to evaluate the model selection procedure described in Section 4. For this purpose we have generated 1000 different data sets/models with $d = 5$ and $N = \{500, 1000\}$ following the same procedure described in the first experiment, but this time we selected the true model to be either a factor model or a DAG with equal probability. In order to generate a factor model, we basically just need to ensure that $\mathbf{D}$ cannot be permuted to a triangular form, so the data generated from it does not admit a DAG representation. We kept 20% of the data to compute the predictive densities to then select between all estimated DAG candidates and the factor model. We found that for $N = 500$ our approach was able to select true DAGs 96.78% of the times and true factor models 87.05%, corresponding to an overall accuracy of 91.9%. Increasing the number of observations, that is, for $N = 1000$, the true DAG, true factor model rates and overall error increased to 98.99%, 95.0% and 96.99%, respectively. Figure 10 shows separately the empirical log-likelihood ratio distributions obtained from the 1000 data sets for DAGs and factor models. The shaded areas correspond to the true DAG/factor model regions, with zero as their boundary. Note that when the wrong model is selected the likelihood ratio is nicely close to the boundary and the overlap of the two distributions decreases with the number of observations used, since the quality of the predictive density increases accordingly. The true DAG rates tend to be larger than for factor models because it is more likely that the latter is confused with a DAG due to estimation errors or closeness to a DAG representation, than a DAG being confused with a factor model which is naturally more general. This is precisely why the likelihood ratios tend to be larger on the factor model side of he plots. All in all, these results demonstrate that our approach is very effective at selecting the true underlying structure when the data is generated by one of the two hypotheses.

6.4 DAGs with Latent Variables

We will start by illustrating the identifiability issues of the model in Equation (1) discussed in Section 2.1 with a very simple example. We generated $N = 500$ observations from the graph in Figure 3(b) and kept 20% of the data to compute test likelihoods. Now, we perform inference on two slightly different models, namely, (u) where $\mathbf{z}' = [z'_1\ z'_2\ z'_L]$ is provided with Laplace distributions with unit variance, that is, $\lambda = 2$, and (i) where z_1, z_2 have Laplace distributions with unit variance and z_L is Cauchy distributed. We want to show that even if both models match the true generating process, (u) is non-identifiable whereas (i) can be successfully estimated. In order to keep the experiment controlled as much as possible, we set $\beta_m = 0.99$ to reflect that the ground truth is dense and we did not infer $\mathbf{C}_D$ and set it to the true values, that is, the identity. Then, we ran 10 independent chains for each one of the models and summarized $\mathbf{B}$, $\mathbf{C}_L$, $\mathbf{D}$ and the test likelihoods in Figure 11.



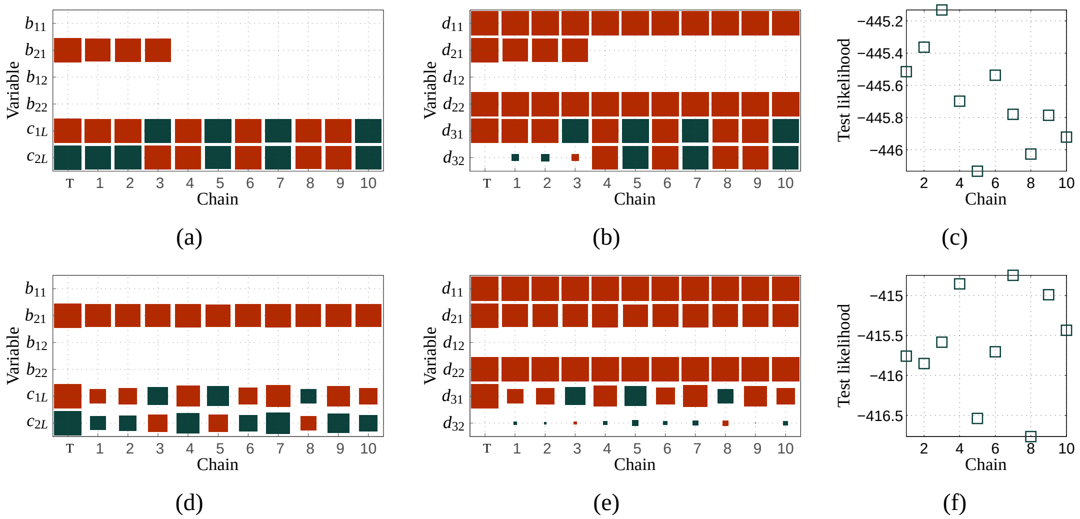
(a) (b)

Figure 10: Log-likelihood ratio empirical distributions for, (a) $N = 500$ and (b) $N = 1000$. Top bars correspond to true factor models, bottom bars to true DAGs and the ratio is computed as described in Section 4. Top bars lying below zero are true factor models predicted to be better explained by DAGs, thus model comparison errors.

Figure 11(a) shows that model (u) finds the DAG in Figure 3(b) (the ground truth) in 3 cases, and in the remaining 7 cases it finds the DAG in Figure 3(a). Note also that the test likelihoods in Figure 11(c) are almost identical, as must be expected due to the lack of identifiability of the model, so they cannot be used to select among the two alternatives. Model (i) finds the right structure all the times as shown in Figure 11(d). The mixing matrix of the equivalent factor model, $\mathbf{D}$ is shown in Figures 11(b) and 11(e) for (u) and (i), respectively. In Figure 11(b), the first and third column of $\mathbf{D}$ exchange positions because all the components of $\mathbf{z}$ have the same distribution, which is not the case of Figure 11(e). The small quantities in $\mathbf{D}$ are due to estimation errors when computing $b_{21}c_{1L} + c_{2L}$, and this cancels out in the true model. The sign changes in Figures 11(a) and 11(d) are caused by the sign ambiguity of $\mathbf{z}_L$ in the product $\mathbf{C}_L\mathbf{z}_L$. We also tested the alternative model in Figure 3(b) obtaining equivalent results, that is, 4 successes for model (u) and 10 for model (i). This small example shows how non-identifiability may lead to two very different DAG solutions with distinct interpretations of the data.

Hoyer et al. (2008) recently presented an approach to DAGs with latent variables based on LiNGAM (Shimizu et al., 2006). Their procedure uses probabilistic ICA and bootstrapping to infer the equivalent factor model distribution $p(\mathbf{D}|\mathbf{X})$, then greedily selects m columns of $\mathbf{D}$ to be latent variables until the remaining ones can be permuted to triangular and the resulting DAG is compatible with the faithfulness assumption (see, Pearl, 2000). If we assume that their procedure is able to find the exact $\mathbf{D}$ for the graphs in Figures 3(a) and 3(b), due to the faithfulness assumption, the DAG in Figure 3(a) will be always selected regardless of the ground truth.[5] In practice, the solution obtained for $\mathbf{D}$ is dense and needs to be pruned, hence we rely on $p(\mathbf{X}, \mathbf{D})$ being larger for the ground truth in Figure 3(b) than for the graph in Figure 3(a), however since both models differ only by a permutation of the columns of $\mathbf{D}$, they have exactly the same joint density $p(\mathbf{X}, \mathbf{D})$—they are non-identifiable, thus the algorithm will select one of the options by chance. Since the source of non-identifiability of their algorithm is permutations of columns of $\mathbf{D}$, it does not matter if probabilistic ICA match or not the distribution of the underlying process as in our model. Anyway, we decided to try models (u)

[5] See Robins et al. (2003) for a very interesting explanation of faithfulness using the same example presented here.



(a) (b) (c)

(d) (e) (f)

Figure 11: Identifiability experiment for the DAG with latent variables. Connectivities **B** and $\mathbf{C}_L$ are shown for (u) in (a) and (i) in (d). Equivalent mixing matrix **D** for (u) in (b) and for (i) in (d). Test likelihoods for (u) and (i) are shown in (c) and (f) respectively. The first column in (a,b,d,e) denoted as T is the ground truth. Dark and light boxes are negative and positive numbers, accordingly.

and (i) described above using the algorithm just described.[6] Regardless of the ground truth, Figures 3(a) or 3(b), the algorithm always selected the DAG in Figure 3(b), which in this particular case is due to $p(\mathbf{X},\mathbf{D})$ being slightly larger for the denser model.

Now we test the model in a more general setting. We generate 100 models and data sets of size $N = 500$ using a similar procedure to the one in the artificial data experiment. The models have $d = 5$ and $m = 1$, no dense structures are generated and the distributions for $\mathbf{z}$ are heavy-tailed, drawn from a generalized Gaussian distribution with random shape. For SLIM, we use the following settings, $\beta_m = 0.1$, $\mathbf{z}_D$ is Laplace with unit variances and $\mathbf{z}_L$ is Cauchy. Furthermore, we have doubled the number of iterations of the DAG sampler, that is, 6000 samples and a burn-in period of 2000, so as to compensate for the additional parameters that need to be inferred due to inclusion of latent variables. Our ordering search procedure was able to find the right ordering 78 out of 100 times. The true positive rates, true negative rates and median AUC are 88.28%, 96.40% and 0.929, respectively, corresponding to approximately 1.5 structure errors per network. Using Hoyer et al. (2008) we obtained 1 true ordering out of 100, 91.63% true positive rate, 65.18% true negative rate and 0.800 median AUC, showing again the preference of the algorithm for denser models. We regard these results as very satisfactory for both methods considering the difficulty of the task and the lack of identifiability of the model by Hoyer et al. (2008).

[6] Matlab package (v.1.1) freely available at `http://www.cs.helsinki.fi/group/neuroinf/lingam/`.

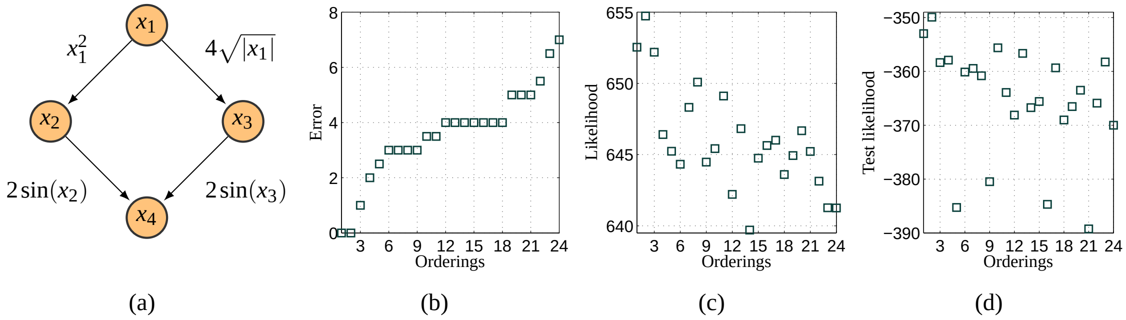


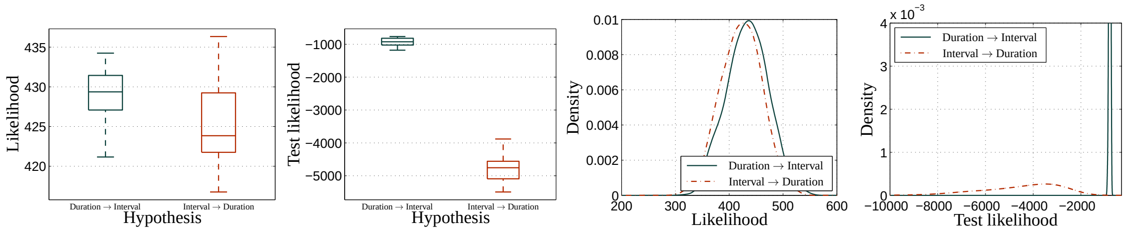
Figure 12: Non-linear DAG artificial example. (a) Network with non-linear interactions between observed nodes used as ground truth. (b,c,d) Median error, likelihood and test likelihood for all possible orderings and 10 independent repetitions. The plots are sorted according to number of errors and only the first two are valid according to the ground truth in (a), that is, $(1,2,3,4)$ and $(1,3,2,4)$. Note that when the error is zero in (b) the likelihoods are larger with respect to the remaining orderings in (c) and (d).

6.5 Non-linear DAGs

For Sparse Non-linear Identifiable Modeling (SNIM) described in Section 3.5, first we want to show that our method can find and select from DAGs with non-linear interactions. We used the artificial network from Hoyer et al. (2009) shown here in Figure 12(a) and generated 10 different data sets corresponding to $N = 100$ observations, each time using driving signals sampled from different heavy-tailed distributions. Since we do not yet have an ordering search procedure for non-linear DAGs, we perform DAG inference for all possible orderings and data sets. The results obtained are evaluated in two ways, first we check if we can find the true connectivity matrix when the ordering is correct. Second, we need to validate that the likelihood is able to select the model with less error and correct ordering among all possible candidates so we can use it in practice. Figures 12(b), 12(c) and 12(d) show the median errors, training and test likelihoods (using 20% of the data) for each one of the orderings, respectively. In this particular case we only have two correct orderings, namely, $(1,2,3,4)$ and $(1,3,2,4)$, corresponding to the first and second candidates in the plots. Figure 12(b) shows that the error is zero only for the two correct orderings, then our model is able to infer the structure once the right ordering is given as desired. As a result of the identifiability, data and test likelihoods shown in Figures 12(c) and 12(d) correlate nicely with the structural error in Figure 12(b). This means that we can use use the likelihoods as a proxy for the structural error just as in the linear case.

We also tested the network in Figure 12(a) using three non-linear structure learning procedures namely greedy standard hill-climbing DAG search, the "ideal parent" algorithm (Elidan et al., 2007) and kernel PC (Tillman et al., 2009). The first two methods use a scaled sigmoid function to capture the non-linearities in the data. In particular, they assume that a variable x can be explained as scaled sigmoid transformation of a linear combination of its parents. The best median result we could obtain after tuning the parameters of the algorithms was 2 errors and 2 reversed links.[7] Both

[7] Maximum number of iterations, random restarts to avoid local minima, regularization of the non-linear regression and the number of ranking candidates in ideal parent algorithm.



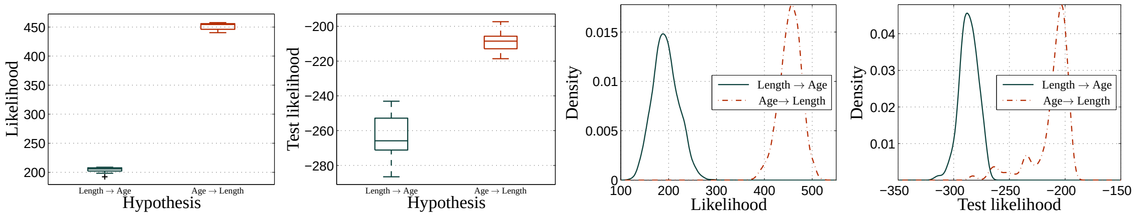
(a) (b) (c) (d)

Figure 13: Testing {duration, interval} in Old Faithful data set. (a,b) Data and test likelihood boxplots for 10 independent repetitions. (c,d) Training and test likelihood densities for one of the repetitions. The test likelihood separates consistently the two tested hypotheses.

methods perform similarly in this particular example, the only significant difference being their computational cost, which is considerably smaller for the "ideal parent" algorithm, as it was also pointed out by Elidan et al. (2007). The reason why we consider these algorithms do not perform well here is that the sigmoid function can be very limited at capturing certain non-linearities due to its parametric form whereas the nonparametric GP gives flexible non-linear functions. The third method uses non-linear independence tests together with non-linear regression (relevance vector machines) and the PC algorithm to produce mixed DAGs. The best median result we could get in this case was 2 errors, 0 reversed links and 1 bidirectional links. These three non-linear DAG search algorithms have the great advantage of not requiring exhaustive enumeration of the orderings as our method and others available in the literature. Zhang and Hyvärinen (2009) provides theoretical evidence of the possibility for flexible non-linear modeling without exhaustive order search but not a way to do it in practice. Yet another possibility not tried here will be to take the best parts of both strategies by taking the outcome of the non-linear DAG search algorithm and refine it using a nonparametric method like SNIM. However, it is not entirely clear how the non-linearities can affect the ordering of the variables. In the remaining part of this section we only focus on tasks for pairs of variables where the ordering search is not an issue.

The data set known as Old Faithful (Asuncion and Newman, 2007) contains 272 observations of two variables measuring waiting time between eruptions and duration of eruptions for the Old Faithful geyser in Yellowstone National Park, USA. We want to test the two possible orderings, duration → interval and interval → duration. Figures 13(a) and 13(b) show training and test likelihood boxplots for 10 independent randomizations of the data set with 20% of the observations used to compute test likelihoods. Our model was able to find the right ordering, that is, duration → interval in all cases when the test likelihood was used but only 7 times with the training likelihood due to the proximity of the densities, see Figure 13(c). On the other hand, the predictive density is very discriminative, as shown for instance in Figure 13(d). This is not a very surprising result since making the duration a function of the interval results in a very non-linear function, whereas the alternative function is almost linear (data not shown).

Abalone is one of the data sets from the UCI ML repository (Azzalini and Bowman, 1990). It is targeted to predict the age of abalones from a set of physical measurements. The data set contains 9 variables and 4177 observations. First we want to test the pair {age, length}. For this purpose, we use 10 subsets of $N = 200$ observations to build the models and compute likelihoods just as



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

Figure 14: Testing {length, age} in Abalone data set. (a,b) Data and test likelihood boxplots for 10 independent repetitions. (c,d) Training and test likelihood densities for one of the repetitions. The likelihoods largely separate the two tested hypotheses.

before. Figures 14(a) and 14(b) show training and test likelihoods respectively as boxplots. Both training and test likelihoods pointed to the right ordering in all 10 repetitions. In this experiment, the separation of the densities for the two hypotheses considered is very large, making age → length significantly better supported by the data. Figures 14(c) and 14(d) show predictive densities for one of the trials indicating again that age → length is consistently preferred. We also decided to try another three sets of hypotheses: {age, diameter}, {age, weight} and {age, length, weight} for which we found the right orderings $\{10,10\}$, $\{10,10\}$ and $\{10,6\}$ out of 10 by looking at the training and the test likelihoods, respectively. In the model with three variables, increasing the number of observations used to fit the model from $N = 200$ to $N = 400$, increased the number of cases in which the test likelihood selected the true hypothesis from 6 to 8 times, which is more than enough to make a decision about the leading hypothesis.

To conclude this set of experiments we test SNIM against another three recently proposed methods,[8] namely Non-linear Additive Noise (NAN) model (Hoyer et al., 2009), Post-Non-Linear (PNL) model (Zhang and Hyvärinen, 2009) and Informational Geometric Causal Inference (IGCI) (Daniusis et al., 2010), using an extended version of "cause-effect pairs" task for the NIPS 2008 causality competition[9] (Mooij and Janzing, 2010). The task consists on distinguishing the cause from the effect of 51 different pairs of observed variables. NAN and PNL rely on an independence test (HSIC, Hilbert-Schmidt Independence Criterion, Gretton et al., 2008) to decide which of the two variable is the cause. NAN was able to take 10 decisions all being accurate. PNL was accurate 40 times out of 42 decisions made. IGCI and SNIM obtained an accuracy of 40 and 39 pairs, respectively.[10] The results indicate (i) that NAN and PNL are very accurate when the independence test used is able to reach a decision and (ii) in terms of accuracy, the results obtained by PNL, IGCI and SNIM are comparable. For SNIM we decide based upon the test likelihood and for IGCI we used a uniform reference measure (rescaling the data between 0 and 1). From the four tested methods we can identify two main trends. One is to explicitly model the data and decide the cause-effect direction using independence tests or test likelihoods like in NAN, PNL and SNIM. The second is to directly define a measure for directionality as in IGCI. The first option has the advantage of being able to convey

[8] Matlab packages available at `http://webdav.tuebingen.mpg.de/causality/`.
[9] Data available at `http://webdav.tuebingen.mpg.de/cause-effect/`.
[10] Results for NAN, PNL and IGCI were taken from Daniusis et al. (2010) because we were unable to entirely reproduce their results with the software provided by the authors.

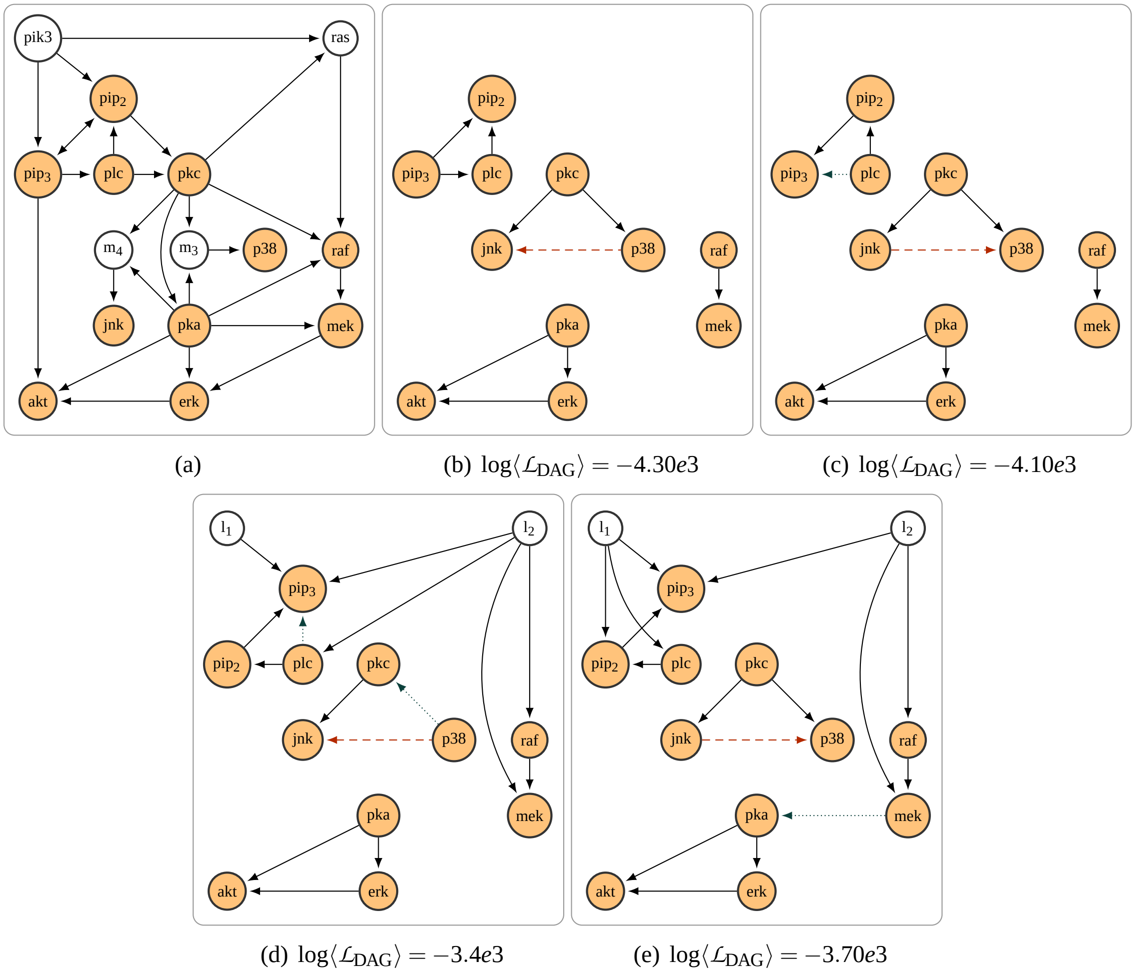
more information about the data at hand whereas the second option is orders of magnitude faster than the other three because it only tests for directionality.

6.6 Protein-signaling Network

This experiment demonstrates a typical application of SLIM in a realistic biological large N, small d setting. The data set introduced by Sachs et al. (2005) consists of flow cytometry measurements of 11 phosphorylated proteins and phospholipids (raf, erk, p38, jnk, akt, mek, pka, pkc, pip$_2$, pip$_3$, plc). Each observation is a vector of quantitative amounts measured from single cells. Data was generated from a series of stimulatory cues and inhibitory interventions. Hence the data is composed of three kinds of perturbations: general activators, specific activators and specific inhibitors. Here we are only using the 1755 observations—clearly non-Gaussian, for example, see Figure 16(a), corresponding to general stimulatory conditions. It is clear that using the whole data set, that is, using specific perturbations, will produce a richer model, however handling interventional data is out of the scope of this paper mainly because handling that kind of data with a factor model is not an easy task. Thus our current order search procedure is not appropriate. Focused only on the observational data, we want to test all the possibilities of our model in this data set, namely, standard factor models, pure DAGs, DAGs with latent variables, non-linear DAGs and quantitative model comparison using test likelihoods. The textbook DAG structure taken from Sachs et al. (see Figure 2 and Table 3, 2005) is shown in Figure 15(a) and the models are estimated using the true ordering and SLIM in Figures 15(b) and 15(c), respectively.

The DAG found using the right ordering of the variables shown in Figure 15(b) turned out to be the same structure found by the discrete Bayesian network from Sachs et al. (2005) without using interventional data (see supplementary material, Figure 4(a)), with one important difference: the method presented by Sachs et al. (2005) is not able to infer the directionality of the links in the graph without interventional data, that is, their resulting graph is undirected. SLIM in Figure 15(c) finds a network almost equal to the one in Figure 15(b) apart from one reversed link, plc → pip3. Surprisingly this was also found reversed by Sachs et al. (2005) using interventional data. In addition, there is just one false positive, the pair $\{$jnk, p38$\}$, even with a dedicated latent variable in the factor model mixing matrix shown in Figure 16(b), thus we cannot attribute such a false positive to estimation errors. A total of 211 ordering candidates were produced during the inference out of approximately 10^7 possible and only $m_{\text{top}} = 10$ of them were used in the structure search step. Note from Figure 16(d) that the predictive densities for the DAGs correlate well with the structural accuracy, apart from candidate 8. Candidates 3 and 8 have the same number of structural errors, however candidate 8 has 3 reversed links instead of 1 as shown in Figure 15(c). The predictive densities for the best candidate, third in Figure 16(d) are shown in Figure 16(c) and suggest that the factor model fits the data better. This makes sense considering that estimated DAG in Figure 15(c) is a substructure of the ground truth. We also examined the estimated factor model in Figure 16(b) and we found that several factors could correspond respectively to three unmeasured proteins, namely pi3k in factors 9 and 11, m$_3$ (mapkkk, mek4/7) and m$_4$ (mapkkk, mek3/6) in factor 7, ras in factors 4 and 6.

We also wanted to assess the performance of our method and several others using this data set, including LiNGAM and those mentioned in the Bayesian network repository experiment, even knowing that this data set contains non-Gaussian data. We found that all of them have similar results in terms of true and false positive rates when comparing them to SLIM. However the number



(a) (b) $\log\langle\mathcal{L}_{\text{DAG}}\rangle = -4.30e3$ (c) $\log\langle\mathcal{L}_{\text{DAG}}\rangle = -4.10e3$

(d) $\log\langle\mathcal{L}_{\text{DAG}}\rangle = -3.4e3$ (e) $\log\langle\mathcal{L}_{\text{DAG}}\rangle = -3.70e3$

Figure 15: Result for protein-signaling network data. (a) Textbook signaling network as reported in Sachs et al. (2005). Estimated structure using SLIM: (b) using the true ordering, (c) obtaining the ordering from the stochastic search, (d) top DAG with 2 latent variables and (e) the runner-up (in test likelihood). False positives are shown in red dashed lines and reversed links in green dotted lines. Below each structure we also report the median test likelihood (larger is better).

of reversed links was not in any case less than 6, which corresponds to more than 50% of the true positives found in every case. This means that they are essentially able to find the skeleton in Figure 15(b). Besides, we do not have knowledge of any other method for DAG learning using only the observational data that also provides results substantially better than the ones shown in Figure 15(c). The poor performance of LiNGAM is difficult to explain but the large amount of reversed links may be due to the FastICA based deterministic ordering search procedure.

We also tried DAG models with latent variables in this data set. The results obtained by the DAG with 2 a priori assumed latent variables are shown in Figures 15(d) and 15(e), corresponding to the first and second DAG candidates in terms of test likelihoods. The first option is different to the pure

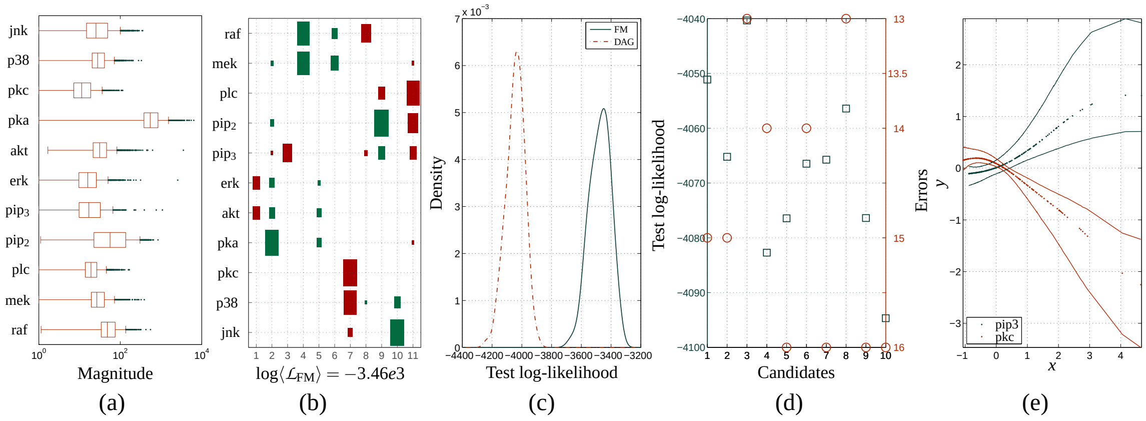


Figure 16: Results for protein-signaling network data. (a) Boxplot for each one of the 11 variables in the data set. (b) Estimated factor model. (c) Test likelihoods for the best DAG (dashed) and the factor model (solid). (d) Test likelihoods (squares) and structure errors (circles) included reversed links for all candidates. (e) Non-linear variables y obtained as a function of the observed variables x for pip3 and pkc. Each dot in the plot is an observation and the solid lines are 95% credible intervals.

DAG in Figure 15(c) only in the reversed link, p38 → pkc, but captures some of the behavior of pik3 and ras in l_1 and l_2 respectively. It is very interesting to see how, due to the link between pik3 and ras that is not possible to model with our model, the second inferred latent variable is detecting signals pointing towards pip$_2$ and plc. We also considered a second option because l_1 in the top model is only connected to a single variable pip$_3$ and thus could be regarded as an estimation error since it can be easily confounded with a driving signal. Comparing Figures 15(c) and 15(e) reveals two differences in the observed part, a false negative pip$_3$ → plc and a new true (reversed) positive mek → pka. This candidate is particularly interesting because the first latent variable captures the connectivity of pik3 while connecting itself to plc due to the lack of connectivity between pip$_3$ and plc. Moreover, the second latent variable resembles ras and the link between pik3 and ras as a link from itself to pip$_3$. In both solutions there is a connection between l_2 and mek that might be explained as a link through a phosphorylation of raf different to the observed one, that is, ras$_{s259}$. In terms of median test likelihoods, the model in Figure 15(d) is only marginally better than the factor model in Figure 16(b) and in turn marginally worse than the DAG in Figure 15(e).

For SNIM we started from the true ordering of the variables but we could not find any improvement compared to the structure in Figure 15(c). In particular there are only two differences, plc → pip$_2$ and jnk → p38 are missing, meaning that at least in this case there are no false positives in the non-linear DAG. Looking at the parameters of the covariance function used, $\boldsymbol{\upsilon}$ (not shown) with acceptance rates of approximately $\approx 20\%$ and reasonable credible intervals, we can say that our model found almost linear functions since all the parameters of the covariance functions are rather small. Figure 16(e) shows two particular non-linear variables learned by the model, corresponding to pip3 and plc. In each case the uncertainty of the estimation nicely increases with the magnitude of the observed variable and although the functions are fairly linear they resemble the saturation effect we can expect in this kind of biological data. From the three non-linear methods non-requiring exhaustive order search described in the previous section (DAG search, "ideal parent" and kPC), the best result we obtained was 11 structural errors, 10 true positives, 34 true negatives, 2 reversed and 6 bidirectional links for kPC vs 12, 9, 34, 1 and 0 by SLIM and 12, 8, 35, 0 and 0 by SNIM.

6.7 Time Series Data

We illustrate the use Correlated Sparse Linear Identifiable Modeling (CLSIM) on the data set introduced by Kao et al. (2004) consisting of temporal gene expression profiles of *E. coli* during transition from glucose to acetate measured using DNA microarrays. Samples from 100 genes were taken at 5, 10, 15, 30, 60 minutes and every hour until 6 hours after transition.[11] The general goal is to reconstruct the unknown transcription factor activities from the expression data and some prior knowledge. In Kao et al. (2004) the prior knowledge consisted of taking the set of transcription factors (ArcA, CRP, CysB, FadR, FruR, GatR, IcIR, LeuO, Lrp, NarL, PhoB, PurB, RpoE, RpoS, TrpR and TyrR) controlling the observed genes and the (up-to-date) connectivity between genes and transcription factors from RegulonDB[12] (Gama-Castro et al., 2008). From this setting, we can immediately relate the transcriptions factors with $\mathbf{Z}$, such a connectivity with $\mathbf{Q}_L$, and their relative strengths with $\mathbf{C}_L$, hence the problem can be seen as a standard factor model. In Kao et al. (2004) they applied a method called Network Component Analysis (NCA), that uses a least-squares based algorithm to solve a problem similar to the one in Equation (1), but assuming that the sparsity pattern (masking matrix $\mathbf{Q}_L$) of $\mathbf{C}_L$ is fixed and known. It is well-known that the information in RegulonDB is still incomplete and hard to obtain for organisms different than *E. coli*. Our goal here is thus to obtain similar transcription factor activities to those found by Kao et al. (2004) without using the information from RegulonDB, but taking into account that the data at hand is a time series by letting each transcription factor activity have an independent Gaussian process prior as described for CSLIM in Section 3.4. We will not attempt to use $\mathbf{Q}_L$ to recover the ground truth connectivity information since RegulonDB is collected from a wide range of experimental conditions and not only from the transcriptional activity produced by the *E. coli* during its transition from glucose to acetate. The results are shown in Figure 17.

Results in Figure 17(e) show the source matrix $\mathbf{Z}$ recovered by our model together with those from NCA.[13] In this experiment we ran a single chain and collected 6000 samples after a burn-in period of 2000 samples (approximately 10 minutes in a desktop machine). Most of the profiles obtained by our method are similar to those obtained by NCA (Kao et al., 2004). We ran two versions of our model, one with $\mathbf{Q}_L$ fixed to the RegulonDB values, that is, similar in spirit to NCA, and another when we infer $\mathbf{Q}_L$ without any restriction. The results of NCA and our model with fixed $\mathbf{Q}_L$ are directly comparable (up to scaling) whereas we had to match the permutation $\mathbf{P}_\mathrm{f}$ of the unrestricted model to those found by NCA in order to compare, using the Hungarian algorithm. Figure 17(a) shows the mixing matrices obtained by NCA and our two models. Figures 17(a) and 17(b) are very similar due to the restriction imposed on $\mathbf{Q}_L$. The mixing matrix obtained by our unrestricted model in Figure 17(c) is clearly denser than the other two, suggesting that there are different ways of connecting genes and transcription factors and still reconstruct the transcription factor activities given the observed gene expression data. When looking to the test log-likelihood

[11] Data available at `http://www.seas.ucla.edu/~liaoj/NCA_module_Data`.

[12] RegulonDB can be found at `http://regulondb.ccg.unam.mx/`.

[13] Matlab package (v.2.3) available at `http://www.seas.ucla.edu/~liaoj/download.htm`.

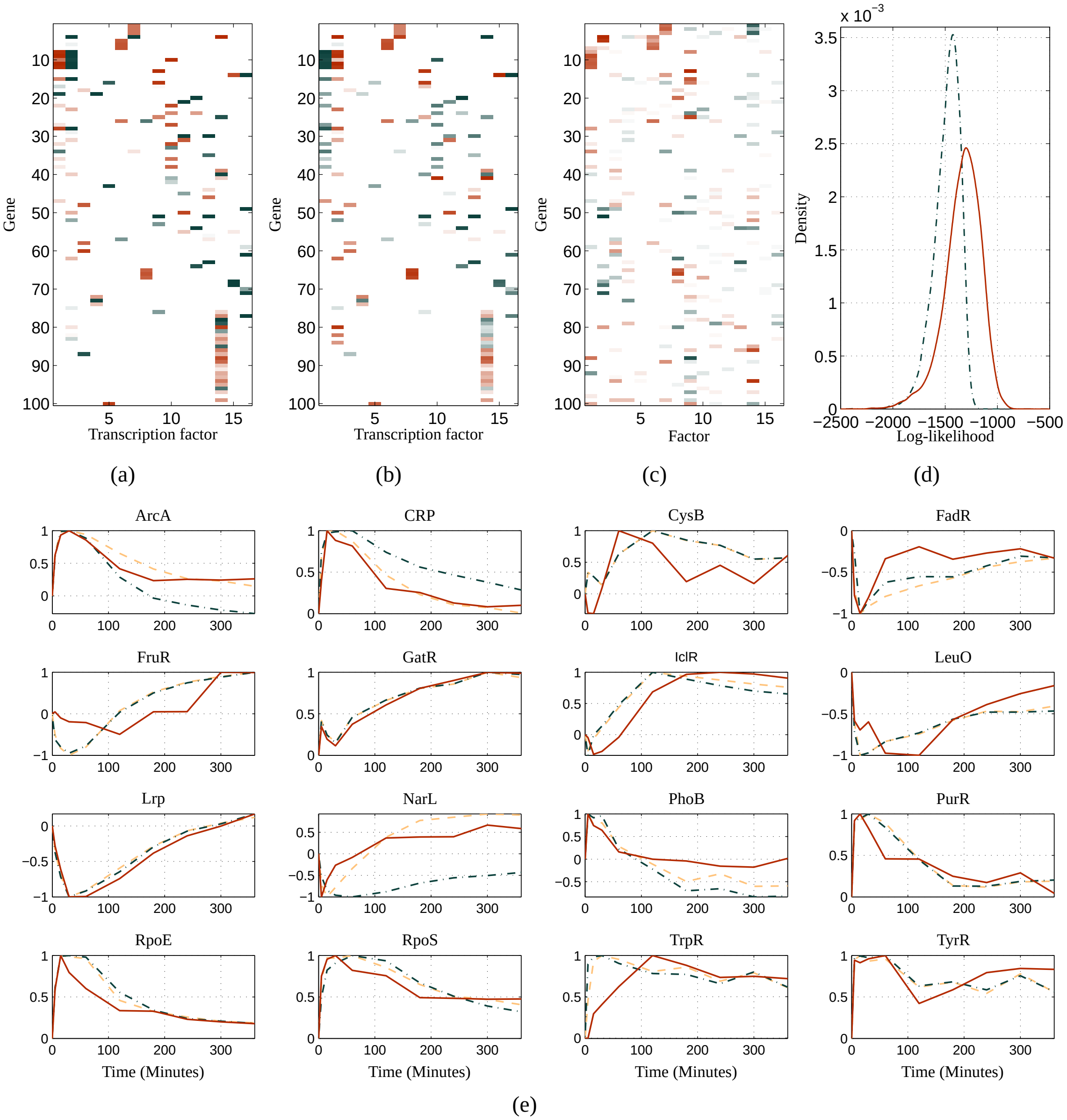


Figure 17: Results for *E. coli* data set. Mixing matrices estimated using: (a) NCA, (b) our formulation when restricting $\mathbf{Q}_L$ using RegulonDB information and (c) the factor model. (d) Model comparison results using test likelihoods. The restricted model (dash-dotted line) obtained a median negative log-likelihood of 1463.4 whereas the unrestricted model (solid line) obtained 1317.1, suggesting no significant model preferences. (e) Estimated transcription factor activities, $\mathbf{Z}$. Our methods (solid and dash-dotted lines for unrestricted and restricted model respectively) produce similar results to those produced by NCA (dashed line).

densities obtained by our two models in Figure 17(d) they are very similar, which suggests that there is no evidence that one of the models makes a better fit on test data. In terms of Mean Squared Error (MSE), NCA obtained 0.0146 while our model reached 0.0264 and 0.0218 on the restricted and unrestricted models, respectively, when using 90% of the data for inference. In addition, the 95% credible intervals for the MSE were $(0.0231, 0.0329)$ and $(0.0164, 0.0309)$ respectively. The latter shows again that there is no evidence that one of the three models is better than the other two, considering that: (i) NCA is trained on the entire data set and (ii) our unrestricted model could, in principle, produce mixing matrices arbitrarily denser than the connectivity matrix extracted from RegulonDB, and thus, again in principle, lower MSE values.

7. Discussion

We have proposed a novel approach called SLIM (Sparse Linear Identifiable Multivariate modeling) to perform inference and model comparison of general linear Bayesian networks within the same framework. The key ingredients for our Bayesian models are slab and spike priors to promote sparsity, heavy-tailed priors to ensure identifiability and predictive densities (test likelihoods) to perform the comparison. A set of candidate orderings is produced by stochastic search during the factor model inference. Subsequently, a linear DAG with or without latent variables is learned for each of the candidates. To the authors' knowledge this is the first time that a method for comparing such closely related linear models has been proposed. This setting can be very beneficial in situations where the prior evidence suggests both DAG structure and/or unmeasured variables in the data. We also show that the DAG with latent variables can be fully identifiable and that SLIM can be extended to the non-linear case (SNIM - Sparse Non-linear Identifiable Multivariate modeling), if the ordering of the variables is provided or can be tested by exhaustive enumeration. For example in the protein-signaling network (Sachs et al., 2005), the textbook ground truth suggests both DAG structure and a number of unmeasured proteins. The previous approach (Sachs et al., 2005) only performed structure learning in pure DAGs but our results using observational data alone suggest that the data is better explained by a (possibly non-linear) DAG with latent variables. Our extensive results on artificial data showed one by one the features of our model in each one of its variants, and demonstrated empirically their usefulness and potential applicability. When comparing against LiNGAM, our method always performed at least as well in every case with a comparable computational cost. The presented Bayesian framework also allows easy extension of our model to match different prior beliefs about the problems at hand without significantly changing the model and its conceptual foundations, as in CSLIM and SNIM.

We believe that the priors that give raise to sparse models in the fully Bayesian inference setting, like the two-level slab (continuous) and spike (point-mass in zero) priors used are very powerful tools for simultaneous model and parameter inference. They may be useful in many settings in machine learning where sparsity of parameters is desirable. Although the posterior distributions for slab and spike priors will be non-convex, it is our experience that inference with blocked Gibbs sampling actually has very good convergence properties. In the two-level approach, one uses a hierarchy of two slab and spike priors. The first is on the parameter and the second is on the mixture parameter (i.e., the probability that the parameter is non-zero). Instead of letting this parameter be controlled by a single Beta-distribution (one level approach) we have a slab and spike distribution on it with a Beta-distributed slab component biased towards one. This makes the model more parsimonious, that is, the probability that parameters are zero or non-zero is closer to zero and one and parameter settings are more robust.

In the following we will discuss open questions and future directions. From the Bayesian network repository experiment it is clear that we need to improve our ordering search procedure if we want to use SLIM for problems with more than say 50 variables. This basically amounts to finding proposal distributions that better exploit the particularities of the model at hand. Another option could be to provide the proposal distribution with some notion of memory to avoid permutations with low probability and/or expand the coverage of the searching procedure.

It is well studied in the literature on sparse models that for increasing number of observations any model tends to loose its sparsity capabilities. This is because the likelihood starts dominating the inference, making the prior distribution less informative. The easiest way to handle such an effect is to make the hyperparameters of the sparsity prior dependent on N. We have not explored this phenomenon in SLIM but it should certainly be taken into account in the specification of sparsity priors.

Directly specifying the distributions of the latent variables in order to obtain identifiability in the general DAG with latent variables requires having different distributions for the driving signals of the observed variables and latent variables. This may introduce model mismatch or be restrictive in some cases as one will not have this kind of knowledge a priori. We thus need more principled ways to specify distributions for $\mathbf{z}$ ensuring identifiably, without restricting some of its components to having a particular behavior, like having heavier tails than the driving signals for instance. We conjecture that providing $\mathbf{z}$ with a parameterization of Dirichlet process priors with appropriate base measures would be enough but we are not certain whether this would be sufficient in practice.

We set a priori that the components of $\mathbf{z}$ are independent. Although this is a very reasonable assumption, it does not allow for connectivity between latent variables as we see for example in the protein signaling network, see Figure 15(a). It is straight forward to specify such a model, although identifiability becomes even harder to ensure in this case.

We do not have an ordering search procedure for the non-linear version of SLIM. This is a necessary step since exhaustive enumeration of all possible orderings is not an option beyond say 10 variables. The main problem is that the non-linear DAG has no equivalent factor model representation so we cannot directly exploit the permutation candidates we find in SLIM. However, as long as the non-linearities are weak, one might in principle use the permutation candidates found in a factor model, that is, the linear effects will determine the correct ordering of the variables.

SLIM cannot handle experimental (interventional) data, and consequently around 80% of the data from the Sachs et al. (2005) study is not used. It is well-established how to learn with interventions in DAGs (see Sachs et al., 2005). The problem remains of how to formulate effective inference with interventional data in the factor model.

Acknowledgments

We thank the editor and the three anonymous referees for their helpful comments and discussions that improved the presentation of this paper. This work is supported by a grant from the Novo Nordisk Foundation to the Bioinformatics Centre, University of Copenhagen.

Appendix A. Gibbs Sampling

Given a set of N observations in d dimensions, the data $\mathbf{X} = [\mathbf{x}_1,\dots,\mathbf{x}_N]$ and m latent variables, MCMC analysis is standard and can be implemented through Gibbs sampling. Note that in the following, $\mathbf{X}_{i:}$ and $\mathbf{X}_{:i}$ are rows and columns of $\mathbf{X}$, respectively, and i, j, n are indexes for dimensions, factors and observations, respectively. In the following we describe the conditional distributions needed to sample from the standard factor model hierarchy. Below we will briefly discus the modifications needed for the DAG.

A.1 Noise Variance

We can sample each element of $\mathbf{\Psi}$ independently using

$$\psi_i^{-1}|\mathbf{X}_{i:},\mathbf{C}_{i:},\mathbf{Z},\mathbf{V}_i,s_s,s_r \;\sim \mathrm{Gamma}\left(\psi_i^{-1}\middle|\, s_s+\frac{N+d}{2}, s_r+u\right) \,, \tag{13}$$

where $\mathbf{V}_i$ is a diagonal matrix with entries τ_{ij} and

$$u = \frac{1}{2}(\mathbf{X}_{i:}-\mathbf{C}_{i:}\mathbf{Z})(\mathbf{X}_{i:}-\mathbf{C}_{i:}\mathbf{Z})^\top + \frac{1}{2}\mathbf{C}_{i:}\mathbf{V}_i^{-1}\mathbf{C}_{i:}^\top \,.$$

A.2 Factors

The conditional distribution of the latent variables $\mathbf{Z}$ using the scale mixtures of Gaussians representation can be computed independently for each element of z_{jn} using

$$z_{jn}|\mathbf{X}_{:n},\mathbf{C}_{:j},\mathbf{Z}_{:n},\mathbf{\Psi},\upsilon_{jn} \;\sim\; \mathcal{N}(z_{jn}|\mathbf{C}_{:j}^\top\mathbf{\Psi}^{-1}\boldsymbol{\varepsilon}_{\setminus jn},u_{jn}) \,, \tag{14}$$

where $u_{jn} = (\mathbf{C}_{:j}^\top\mathbf{\Psi}^{-1}\mathbf{C}_{:j}+\upsilon_{jn}^{-1})^{-1}$ and $\boldsymbol{\varepsilon}_{\setminus jn} = \mathbf{X}_{:n} - \mathbf{C}\mathbf{Z}_{:n}|_{z_{jn}=0}$. If the latent factors are Laplace distributed the mixing variances υ_{jn} have exponential distribution, thus the resulting conditional is

$$\upsilon_{jn}^{-1}|z_{jn},\lambda \;\sim \mathrm{IG}\left(\upsilon_{jn}^{-1}\middle|\,\frac{\lambda}{|z_{jn}|},\lambda^2\right) \,,$$

and for the Student's t, with corresponding gamma densities as

$$\upsilon_{jn}^{-1}|z_{jn},\sigma^2,\theta \;\sim \mathrm{Gamma}\left(\upsilon_{jn}^{-1}\middle|\,\frac{\theta+1}{2},\frac{\theta}{2}+\frac{z_{jn}^2}{2\sigma^2}\right) \,,$$

where $\mathrm{IG}(\cdot|\mu,\lambda)$ is the inverse Gaussian distribution with mean μ and scale parameter λ (Chhikara and Folks, 1989).

A.3 Gaussian Processes

In practice, the prior distribution for each row of the matrix $\mathbf{Z}$ in CSLIM has the form $z_{j1},\dots,z_{jN} \sim \mathcal{N}(0,\mathbf{K}_j)$, where $\mathbf{K}_j$ is a covariance matrix of size $N\times N$ built using $k_{\upsilon_{j,n}}(n,n')$. The conditional distribution for $z_{j1},\dots,z_{jN}$ can be computed using

$$z_{j1},\dots,z_{jN}|\mathbf{X},\mathbf{C}_{j:},\mathbf{Z}_{\setminus j},\mathbf{\Psi} \;\sim\; \mathcal{N}(z_{j1},\dots,z_{jN}|\mathbf{C}_{:j}^\top\mathbf{\Psi}^{-1}\boldsymbol{\varepsilon}_{\setminus j}\mathbf{V},\mathbf{V}) \,,$$

where $\mathbf{Z}_{\setminus j}$ is $\mathbf{Z}$ without row j, $\mathbf{V} = (\mathbf{U}+\mathbf{K}_j^{-1})^{-1}$, $\mathbf{U}$ is a diagonal matrix with elements $\mathbf{C}_{:j}^{\top}\boldsymbol{\Psi}^{-1}\mathbf{C}_{:j}$ and $\boldsymbol{\varepsilon}_{\setminus j} = \mathbf{X}-\mathbf{C}\mathbf{Z}|_{z_{j1},\ldots,z_{jN}=0}$. The computation of $\mathbf{V}$ can be done in a numerically stable way by rewriting $\mathbf{V} = \mathbf{K}_j - \mathbf{K}_j(\mathbf{U}^{-1}+\mathbf{K}_j)^{-1}\mathbf{K}_j$ and then using Cholesky decomposition and back substitution to obtain in turn $\mathbf{L}\mathbf{L}^{\top} = \mathbf{U}^{-1}+\mathbf{K}_j$ and $\mathbf{L}^{-1}\mathbf{K}_j$. The hyperparameters of the covariance function in Equation (9) can be sampled using

$$\kappa|\boldsymbol{\upsilon},k_s,k_r \;\sim\; \text{Gamma}\left(\kappa\middle|k_s+mu_s,k_r+\sum_{j=1}^{m}\upsilon_j\right) .$$

For the inverse length-scales we use Metropolis-Hastings updates with proposal $q(\upsilon_j^{\star}|\upsilon_j) = p(\upsilon_j^{\star})$ and acceptance ratio

$$\xi_{\to\star} = \frac{\mathcal{N}(z_{j1},\ldots,z_{jN}|\mathbf{0},\mathbf{K}_j^{\star})}{\mathcal{N}(z_{j1},\ldots,z_{jN}|\mathbf{0},\mathbf{K}_j)} ,$$

where $\mathbf{K}_j^{\star}$ is obtained using $k_{\upsilon_j^{\star},n}(n,n')$. For SNIM, we only need to replace $\mathbf{C}$ by $\mathbf{B}$, $\mathbf{Z}$ by $\mathbf{Y} = [\mathbf{y}_1\ \ldots\mathbf{y}_N]$ and $k_{\upsilon_j,n}(n,n')$ by $k_{\upsilon_i,x}(\mathbf{x},\mathbf{x}')$.

A.4 Mixing Matrix

In order to sample each c_{ij} from the conditional distribution of the matrix $\mathbf{C}$ we use

$$c_{ij}|\mathbf{X}_{i:},\mathbf{C}_{\setminus ij},\mathbf{Z}_{j:},\psi_i,\tau_{ij} \;\sim\; \mathcal{N}(c_{ij}|u_{ij}\boldsymbol{\varepsilon}_{\setminus ij}\mathbf{Z}_{j:}^{\top},u_{ij}\psi_i) , \qquad (15)$$

where $u_{ij} = (\mathbf{Z}_{j:}\mathbf{Z}_{j:}^{\top}+\tau_{ij}^{-1})^{-1}$ and $\boldsymbol{\varepsilon}_{\setminus ij} = \mathbf{X}_{i:}-\mathbf{C}_{i:}\mathbf{Z}|_{d_{ij}=0}$. Note that we only need to sample those c_{ij} for which $r_{ij}=1$, that is, just the slab distribution. Sampling from the conditional distributions for τ_{ij} can be done using

$$\tau_{ij}^{-1}|d_{jn},t_s,t_r \;\sim \text{Gamma}\left(\tau_{ij}^{-1}\middle|t_s+\frac{1}{2},t_r+\frac{d_{ij}^2}{2\psi_i}\right) . \qquad (16)$$

The conditional distributions for the remaining parameters in the slab and spike prior can be written first for the masking matrix $\mathbf{Q}$ as

$$q_{ij}|\mathbf{X}_{i:},\mathbf{D}_{i:},\mathbf{Z},\psi_i,\tau_{ij},\eta_{ij} \;\sim \text{Bernoulli}\left(q_{ij}\middle|\frac{\xi_{\eta_{ij}}}{1+\xi_{\eta_{ij}}}\right) , \qquad (17)$$

where

$$\xi_{\eta_{ij}} \;=\; \frac{\alpha_m\nu_j}{1-\alpha_m\nu_j}\frac{\psi_i^{1/2}}{(\mathbf{Z}_{j:}\mathbf{Z}_{j:}^{\top}+\tau_{ij}^{-1})^{1/2}}\exp\left(\frac{(\boldsymbol{\varepsilon}_{\setminus ij}\mathbf{Z}_{j:}^{\top})^2}{2\psi_i(\mathbf{Z}_{j:}\mathbf{Z}_{j:}^{\top}+\tau_{ij}^{-1})}\right) ,$$

and the probability of each element of $\mathbf{C}$ of being non-zero as

$$\eta_{ij}|u_{ij},q_{ij},\alpha_p,\alpha_m \;\sim (1-u_{ij})\delta(\eta_{ij})+u_{ij}\text{Beta}(\eta_{ij}|\alpha_p\alpha_m+q_{ij},\alpha_p(1-\alpha_m)+1-q_{ij}) , \qquad (18)$$

where $u_{ij} \sim \text{Bernoulli}(h_{ij}|r_{ij}+(1-r_{ij})\nu_j(1-\alpha_m)/(1-\nu_j\alpha_m))$, that is, we set $u_{ij}=1$ if $q_{ij}=1$. Finally, for the column-wise shared sparsity rate we have

$$\nu_j|\mathbf{u}_j,\beta_p,\beta_m \;\sim \text{Beta}\left(\nu_j\middle|\beta_p\beta_m+\sum_{i=1}^{d}u_{ij},\beta_p(1-\beta_m)+\sum_{i=1}^{d}(1-u_{ij})\right) . \qquad (19)$$

Sampling from the DAG model only requires minor changes in notation but the conditional posteriors are essentially the same. The changes mostly amount to replacing accordingly $\mathbf{C}$ by $\mathbf{B}$ and $\mathbf{Q}$ by $\mathbf{R}$. Note that $\mathbf{Q}_L$ is the identity and $\mathbf{R}$ is strictly lower triangular a priori, thus we only need to sample their active elements.

A.5 Inference with Missing Values

We introduce a binary masking matrix indicating whether an element of $\mathbf{X}$ is missing or not. For the factor model we have the following modified likelihood

$$p(\mathbf{X}_{\text{tr}}|\mathbf{C},\mathbf{Z},\boldsymbol{\Psi},\mathbf{M}_{\text{miss}}) = \mathcal{N}(\mathbf{M}_{\text{miss}} \odot \mathbf{X}|\mathbf{M}_{\text{miss}} \odot (\mathbf{C}\mathbf{Z}),\boldsymbol{\Psi}) \ .$$

Testing on the missing values, $\mathbf{M}^{\star}_{\text{miss}} = \mathbf{1}\mathbf{1}^{\top} - \mathbf{M}$ requires averaging the test likelihood

$$p(\mathbf{X}^{\star}|\mathbf{C},\mathbf{Z},\boldsymbol{\Psi},\mathbf{M}^{\star}_{\text{miss}}) = \mathcal{N}(\mathbf{M}^{\star}_{\text{miss}} \odot \mathbf{X}|\mathbf{M}^{\star}_{\text{miss}} \odot (\mathbf{C}\mathbf{Z}),\boldsymbol{\Psi}) \ ,$$

over $\mathbf{C},\mathbf{Z},\boldsymbol{\Psi}$ given $\mathbf{X}_{\text{tr}}$ (training). We can approximate the predictive density $p(\mathbf{X}^{\star}|\mathbf{X}_{\text{tr}},\cdot)$ by computing the likelihood above during sampling using the conditional posteriors of $\mathbf{C}$, $\mathbf{Z}$ and $\boldsymbol{\Psi}$ and then summarizing using for example the median. Drawing from $\mathbf{C}$, $\mathbf{Z}$, $\boldsymbol{\Psi}$ can be achieved by sampling from their respective conditional distributions as described before with some minor modifications.

References

D. F. Andrews and C. L. Mallows. Scale mixtures of normal distributions. *Journal of the Royal Statistical Society: Series B (Methodology)*, 36(1):99–102, 1974.

A. Asuncion and D .J. Newman. UCI machine learning repository, 2007.

A. Azzalini and A. W. Bowman. A look at some data on the Old Faithful geyser. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 39(3):357–365, 1990.

P. Bekker and J. M. F. ten Berge. Generic global indentification in factor analysis. *Linear Algebra and its Applications*, 264(1–3):255–263, 1997.

M. Branco and D. K. Dey. A general class of multivariate skew-elliptical distributions. *Journal of Multivariate Analysis*, 79(1):99–113, 2001.

C. M. Carvalho, J. Chang, J. E. Lucas, J. R. Nevins, Q. Wang, and M. West. High-dimensional sparse factor modeling: Applications in gene expression genomics. *Journal of the American Statistical Association*, 103(484):1438–1456, 2008.

R. S. Chhikara and L. Folks. *The Inverse Gaussian Distribution: Theory, Methodology, and Applications*. M. Dekker, New York, 1989.

S. Chib. Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, 90(732):1313–1321, 1995.

D. M. Chickering. Learning Bayesian networks is NP-complete. In D. Fisher and H.-J. Lenz, editors, *Learning from Data: AI and Statistics*, pages 121–130. Springer-Verlag, 1996.

P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36(3):287–314, 1994.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

P. Daniusis, J. Janzing, J. Mooij, J. Zscheischler, B. Steudel, K. Zhang, and B. Schölkopf. Inferring deterministic causal relations. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 2010.

A. P Dawid and S. L Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Annals of Statistics*, 21(3):1272–1317, 1993.

A. P. Dempster. Covariance selection. *Biometrics*, 28:157–175, 1972.

G. Elidan, I. Nachman, and N. Friedman. “Ideal Parent” structure learning for continuous variable Bayesian networks. *Journal of Machine Learning Research*, 8:1799–1833, 2007.

N. Friedman and D. Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1–2):95–125, 2003.

N. Friedman and I. Nachman. Gaussian process networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 211–219. 2000.

N. Friedman, I. Nachman, and D. Pe’er. Learning Bayesian network structure from massive datasets: The “sparse candidate” algorithm. In K. B. Laskey and H. Prade, editors, *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 206–215, 1999.

N. Friel and A. N. Pettitt. Marginal likelihood estimation via power posteriors. *Journal of the Royal Statistical Society: Series B (Methodology)*, 70(3):589–607, 2008.

S. Gama-Castro, V. Jiménez-Jacinto, M. Peralta-Gil, A. Santos-Zavaleta, M. I. Peñaloza-Spinola, B. Contreras-Moreira, J. Segura-Salazar, L. Muñiz-Rascado, I. Martínez-Flores, H. Salgado, C. Bonavides-Martínez, C. Abreu-Goodger, C. Rodríguez-Penagos, J. Miranda-Ríos, E. Morett, E. Merino, A. M. Huerta, L. Treviño-Quintanilla, and J. Collado-Vides. RegulonDB (version 6.0): gene regulation model of Escherichia coli K-12 beyond transcription, active (experimental) annotated promoters and textpresso navigation. *Nucleic Acids Research*, 36(Database Issue): 120–124, 2008.

E. I. George and R. E. McCulloch. Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423):881–889, 1993.

J. Geweke. Variable selection and model comparison in regression. In J. Berger, J. Bernardo, A. Dawid, and A. Smith, editors, *Bayesian Statistics 5*, pages 609–620. Oxford University Press, 1996.

Z. Ghahramani, T. L. Griffiths, and P. Sollich. Bayesian nonparametric latent feature models. In J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, editors, *Bayesian Statistics 8*, pages 201–226. Oxford University Press, 2006.

P. Giudici and P. J Green. Decomposable graphical Gaussian model determination. *Biometrika*, 86 (4):785–801, 1999.

A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 585–592. MIT Press, 2008.

D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1:49–75, 2000.

R. Henao and O. Winther. Bayesian sparse factor models and DAGs inference and comparison. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 736–744. The MIT Press, 2009.

P. O. Hoyer, S. Shimizu, A. J. Kerminen, and M. Palviainen. Estimation of causal effects using linear non-Gaussian causal models with hidden variables. *Interantional Journal of Approximate Reasoning*, 49(2):362–378, 2008.

P .O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 689–696. 2009.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley-Interscience, 2001.

H. Ishwaran and J. S. Rao. Spike and slab variable selection: Frequentist and Bayesian strategies. *Annals of Statistics*, 33(2):730–773, 2005.

I. T. Jolliffe, N. T. Trendafilov, and M. Uddin. A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics*, 12(3):531–547, 2003.

A. M. Kagan, YU. V Linnik, and C. Radhakrishna Rao. *Characterization Problems in Mathematical Statistics*. Probability and Mathematical Statistics. Wiley, New York, 1973.

K. C. Kao, Y-L. Yang, R. Boscolo, C. Sabatti, V. Roychowdhury, and J. C. Liao. Transcriptome-based determination of multiple transcription regulator activities in Escherichia Coli by using network component analysis. *PNAS*, 101(2):641–646, 2004.

D. Knowles and Z. Ghahramani. Infinite sparse factor analysis and infinite independent components analysis. In M. E. Davies, C. C. James, S. A. Abdallah, and M. D. Plumbley, editors, *7th International Conference on Independent Component Analysis and Signal Separation*, volume 4666 of *Lecture Notes in Computer Science*, pages 381–388. Springer-Verlag, Berlin, 2007.

F. B. Lempers. *Posterior Probabilities of Alternative Linear Models*. Rotterdam University Press, 1971.

H. F. Lopes and M. West. Bayesian model assessment in factor analysis. *Statistica Sinica*, 14(1): 41–67, 2004.

J. Lucas, C. Carvalho, Q. Wang, A. Bild, J. R. Nevins, and M. West. *Bayesian Inference for Gene Expression and Proteomics*, chapter Sparse Statistical Modeling in Gene Expression Genomics, pages 155–176. Cambridge University Press, 2006.

J. K. Martin and R. P. McDonald. Bayesian estimation in unrestricted factor analysis: A treatment for heywood cases. *Psychometrika*, 40(4):505–517, 1975.

T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

J. Mooij and D. Janzing. Distinguishing between cause and effect. In *JMLR Workshop and Conference Proceedings*, volume 6, pages 147–156, 2010.

I. Murray. *Advances in Markov Chain Monte Carlo Methods*. PhD thesis, Gatsby computational neuroscience unit, University College London, 2007.

R. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

T. Park and G. Casella. The Bayesian lasso. *Journal of the American Statistical Association*, 103 (482):681–686, 2008.

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

P. Rai and H. Daume III. The infinite hierarchical factor regression model. In D. Koller, D. Schurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1321–1328. The MIT Press, 2009.

B. Rajaratman, H. Massam, and C. Carvalho. Flexible covariance estimation in graphical gaussian models. *Annals of Statistics*, 36(6):2818–2849, 2008.

J. M. Robins, R. Scheines, P. Spirtes, and L. Wasserman. Uniform consistency in causal inference. *Biometrika*, 90(3):491–515, 2003.

K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

M. W. Schmidt, A. Niculescu-Mizil, and K. P. Murphy. Learning graphical model structure using L1-regularization paths. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, pages 1278–1283, 2007.

S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

R. Silva. *Causality in the Sciences*, chapter Measuring Latent Causal Structure. Oxford University Press, 2010.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. The MIT Press, second edition, 2001.

M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, pages 548–549, 2005.

R. Thibaux and M. I. Jordan. Hierarchical beta processes and the indian buffet process. In M. Meila and X. Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, pages 564–571, 2007.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodology)*, 58(1):267–288, 1996.

R. Tillman, A. Gretton, and P. Spirtes. Nonlinear directed acyclic structure learning with weakly additive noise models. In *Advances in Neural Information Processing Systems 22*, pages 1847–1855. Y. Bengio and D. Schuurmans and J. Lafferty and C. K. I. Williams and A. Culotta, 2009.

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.

M. West. On scale mixtures of normal distributions. *Biometrika*, 74(3):646–648, 1987.

M. West. Bayesian factor regression models in the “large *p*, small *n*” paradigm. In J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, editors, *Bayesian Statistics 7*, pages 723–732. Oxford University Press, 2003.

S. Yu, V. Tresp, and K. Yu. Robust multi-task learning with *t*-processes. In *Proceedings of the 24th International Conference on Machine Learning*, volume 227, pages 1103–1110, 2007.

K. Zhang and A. Hyvärinen. On the identifiability of the post-nonlinear causal model. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pages 647–655. AUAI Press, 2009.

K. Zhang and A. Hyvärinen. Distinguishing causes from effect using nonlinear acyclic causal models. In *JMLR Workshop and Conference Proceedings*, volume 6, pages 157–164, 2010.

H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):262–286, 2006.