Partially non-homogeneous dynamic Bayesian networks based on Bayesian regression models with partitioned design matrices
Shafiee Kamalabad, Mahdi; Heberle, Alexander Martin; Thedieck, Kathrin; Grzegorczyk, Marco

Published in:
Bioinformatics (Oxford, England)

DOI:
10.1093/bioinformatics/bty917

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2019

Link to publication in University of Groningen/UMCG research database

Citation for published version (APA):

Copyright
Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

Take-down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): http://www.rug.nl/research/portal. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.
Systems biology

**Partially non-homogeneous dynamic Bayesian networks based on Bayesian regression models with partitioned design matrices**

Mahdi Shafiee Kamalabad¹, Alexander Martin Heberle², Kathrin Thedieck²,³ and Marco Grzegorczyk¹,*

¹Department of Mathematics, Bernoulli Institute, Faculty of Science and Engineering, University of Groningen, 9747 AG Groningen, The Netherlands, ²Laboratory of Pediatrics, Section Systems Medicine of Metabolism and Signaling, University of Groningen, University Medical Center Groningen, 9713 AV Groningen, The Netherlands and ³Department for Neuroscience, School of Medicine and Health Sciences, Carl von Ossietzky University Oldenburg, 26129 Oldenburg, Germany

*To whom correspondence should be addressed.

Associate Editor: Jonathan Wren

Received on March 28, 2018; revised on August 29, 2018; editorial decision on October 12, 2018; accepted on November 2, 2018

**Abstract**

**Motivation:** Non-homogeneous dynamic Bayesian networks (NH-DBNs) are a popular modelling tool for learning cellular networks from time series data. In systems biology, time series are often measured under different experimental conditions, and not rarely only some network interaction parameters depend on the condition while the other parameters stay constant across conditions. For this situation, we propose a new partially NH-DBN, based on Bayesian hierarchical regression models with partitioned design matrices. With regard to our main application to semi-quantitative (immunoblot) timecourse data from mammalian target of rapamycin complex 1 (mTORC1) signalling, we also propose a Gaussian process-based method to solve the problem of non-equidistant time series measurements.

**Results:** On synthetic network data and on yeast gene expression data the new model leads to improved network reconstruction accuracies. We then use the new model to reconstruct the topologies of the circadian clock network in *Arabidopsis thaliana* and the mTORC1 signalling pathway. The inferred network topologies show features that are consistent with the biological literature.

**Availability and implementation:** All datasets have been made available with earlier publications. Our Matlab code is available upon request.

**Contact:** m.a.grzegorczyk@rug.nl

**Supplementary information:** Supplementary data are available at *Bioinformatics* online.

---

**1 Introduction**

Dynamic Bayesian networks (DBNs) have become a popular tool for learning the topologies of cellular regulatory networks from time series data. The classical (homogeneous) DBN models assume that the network parameters stay constant in time, so that the network structure is inferred along with one single set of network parameters (Friedman et al., 2000). Many regulatory processes are non-stationary so that this homogeneity assumption is too restrictive. To allow for time-dependent parameters, many authors have proposed to combine DBNs with multiple changepoint processes (Ahmed and Xing, 2009; Dondelinger et al., 2013; Grzegorczyk and Husmeier, 2013; Husmeier et al., 2010; Lèbre et al., 2010; Robinson and Hartemink, 2010) or with hidden Markov models (Grzegorczyk, 2016). In those models a multiple changepoint process (or a hidden Markov model) divides the temporal data into disjoint components with component-specific network parameters. The network
structure, the data segmentation and the component-specific network parameters are inferred from the data. Those models are often referred to as non-homogeneous DBNs (NH-DBNs).

In many real-world applications, in particular in systems biology, data are often collected under different experimental conditions. That is, instead of one single (long) time series that has to be segmented, there is K (short) time series. The data are then intrinsically divided into K unordered components, and there is no need for inferring the segmentation. In this situation, it is normally not clear a priori whether the network parameters stay constant across components (conditions) or whether they vary from component to component (with the conditions). Three biological systems that we will consider in this article are: Section 4.4: The parameters of a mTORC1 protein signalling network can change in the presence of cycles, to which the plants were earlier exposed. Section 4.6: The parameters of the circadian clock can depend on the dark:light ratio, which the plants were earlier exposed to. As the same regression model is applied to each Z_t separately, we describe it using a general notation, where Y is the response and X_1, ..., X_n are the covariates.

2 Materials and methods

DBNs and NH-DBNs are used to infer networks showing the regulatory interactions among variables Z_1, ..., Z_N. The interactions are subject to a time lag, so that there is no need for an acyclic network structure. Hence, dynamic network inference can be thought of as inferring the covariate sets for N independent regression models. In the ith model, Z_i is the response and the remaining X_j := N - i variables Z_{i+1}, ..., Z_N at time point t - 1 are used as potential covariates for Z_i at time point t. The goal is to infer a covariate set for each Z_i and the system of covariate sets describes a network; see Section 2.6 for details. As the same regression model is applied to each Z_t separately, we describe it using a general notation, where Y is the response and X_1, ..., X_n are the covariates.

2.1 Bayesian regression with partitioned design matrix

We consider a regression model with response Y and covariates X_1, ..., X_n. We assume that data were measured under K experimental conditions, which we refer to as K components. We further assume that the data for each component k ∈ {1, ..., K} were measured at equidistant time points t = 1, ..., T_k. Let y_{k,t} and x_{k,i,t} denote the values of Y and X_i at the ith time point of component k. In dynamic networks, the interactions are subject to a time lag \Theta, which is usually set to one time point. That is, the values x_{1,k}, ..., x_{n,k} correspond to the response value y_{k,t+1}. For each component k we build a component-specific response vector y_k and the corresponding design matrix X_k, where X_k includes a first column of 1’s for the intercept:

\[ y_k = (y_{k,1}, \ldots, y_{k,T_k})^T, \quad X_k = (1 \quad x_{1,k} \quad \ldots \quad x_{n,k}) \]

where \[ x_{i,k} = (x_{i,k,1}, \ldots, x_{i,k,T_k})^T \]

For each k we could assume a separate Gaussian likelihood:

\[ y_k \sim \mathcal{N}_{T_k-1}(X_k \beta_k, \sigma_k^2 I) \quad (k = 1, \ldots, K) \]

where I is the identity matrix, \[ \beta_k = (\beta_{k,1}, \beta_{k,2}, \ldots, \beta_{k,n})^T \]

is the component-specific vector of regression coefficients, and \[ \sigma_k^2 \]

is the component-specific noise variance. Imposing independent priors on each pair \( \{ \beta_k, \sigma_k^2 \} \), leads to K independent models. Alternatively, we could merge the data \[ y := (y_1^T, \ldots, y_K^T)^T \]

and \[ X := (X_1^T, \ldots, X_K^T)^T \]

and employ one model for the merged data:

\[ y \sim \mathcal{N}_T(X \beta, \sigma^2 I) \quad (T := \sum_{k=1}^{K} (T_k - 1)) \]

so that \[ \beta = (\beta_0, \beta_1, \ldots, \beta_n)^T \] would apply to all components. When some covariates have a component-specific and other covariates have a constant regression coefficient, both likelihoods (2) and (3) are suboptimal. For this situation, we propose a new partially non-homogeneous regression model that infers the best trade-off from the data. The key idea is to use a likelihood with a partitioned design matrix.

For now, we assume that we know for each coefficient whether it is component-specific or constant. Let the intercept and the first n_1 < n coefficients stay constant while the remaining n_2 = n - n_1 coefficients are component-specific. We then have the regression equation:

\[ y_{k,t+1} = \beta_0 + \sum_{i=1}^{n_1} \beta_i \cdot x_{i,k,t} + \sum_{i=n_1+1}^{n} \beta_{n_1+i} \cdot x_{i,k,t} + \epsilon_{k,t+1} \]

where \[ \epsilon_{k,t+1} \sim \mathcal{N}(0, \sigma^2) \], and the likelihood takes the form:

\[ y \sim \mathcal{N}_T(X \beta_0, \sigma^2 I) \]

(4)

where \[ \beta_0 \] is a vector of \( (1 + n_1 + K - n_2) \) regression coefficients, and \[ X_0 \] is a partitioned matrix with \[ T = \sum_{k=1}^{K} (T_k - 1) \] rows and \( (1 +
where $x_{k} = (x_{k,2}, \ldots, x_{k, T_{k}-1})^T$, and $\beta_B$ is of the form:

$$((\beta_{0}, \beta_{1}, \ldots, \beta_{n_{0}}), (\beta_{b_{1,1}}, \ldots, \beta_{b_{n_{1}}}), (\beta_{b_{2,1}}, \ldots, \beta_{b_{n_{2}}}))^T$$

The first subvector of $\beta_B$ is the vector $\beta_k := (\beta_{0}, \beta_{1}, \ldots, \beta_{n_{0}})^T$ of the regression coefficients that stay constant, and then there is a subvector $\beta_k := (\beta_{b_{1,1}}, \ldots, \beta_{b_{n_{1}}})^T$ for each component $k$ with the component-specific coefficients. For the noise variance parameter $\sigma^2$ we use an inverse Gamma prior, $\sigma^{-2} \sim \text{Gam}(a, b)$, and on $\beta$, we impose a Gaussian prior with zero mean vector:

$$\beta \sim \mathcal{N}_{n_0}(0, \sigma^2 I_{n_0})$$

(5)

For the component-specific vectors $\beta_1, \ldots, \beta_K$ we adapt the idea from Grzegorczyk and Husmeier (2013), and impose a hyperprior:

$$\beta_k \sim \mathcal{N}_{n_0}(\mu, \sigma^2 I_{n_0}) \quad (k = 1, \ldots, K) \quad \text{and} \quad \mu \sim \mathcal{N}_{n_0}(\mu_0, \Sigma_0)$$

(6)

The hyperprior couples the vectors $\beta_1, \ldots, \beta_K$ hierarchically and encourages them to stay similar across components. Re-using the variance parameter $\sigma^2$ in (5 and 6) allows us to incorporate the regression coefficients vectors and the noise variance to be integrated out in the likelihood, i.e., the marginal likelihood $p(\mathbf{y}|\mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu)$ to be computed analytically (see below). For $\mathbf{x}_{\mathcal{L}}$ and $\mathbf{\lambda}_{\mathcal{L}}$ we also use inverse Gamma priors:

$$\mathbf{x}_{\mathcal{L}} \sim \text{Gam}(\mathbf{a}, \mathbf{b}) \quad \text{and} \quad \mathbf{\lambda}_{\mathcal{L}} \sim \text{Gam}(\mathbf{\alpha}, \mathbf{\beta})$$

The prior of $\beta_{B}$ is a product of Gaussians:

$$p(\beta_B|\sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu) = p(\beta_{0}|\sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu) \cdot \prod_{k=1}^{K} p(\beta_k|\sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu)$$

Given $\sigma^2$, $\mathbf{x}_{\mathcal{L}}$, $\mathbf{\lambda}_{\mathcal{L}}$, and $\mu$, the Gaussians are independent, so that:

$$\beta_{B} | (\sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu) \sim \mathcal{N}_{1+n_0+r_{\mathcal{L}}}(\mu, \sigma^2 \Sigma)$$

with $\mu = (\mu^T, \mu^T, \ldots, \mu^T)^T$ and $\Sigma = \left( \begin{array}{ccc} \Sigma & 0 & \vdots \\ 0 & \Sigma & \vdots \\ \vdots & \vdots & \ddots \end{array} \right)$

where $L$ is the $(n_0 + 1)$-dimensional and $C_3$ the $(K \cdot n_2)$-dimensional identity matrix. We have for the posterior distribution:

$$p(\beta_B, \sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu|\mathbf{y}) \propto p(\mathbf{y}|\sigma^2, \beta_B) \cdot p(\mathbf{y}|\mathbf{y}, \beta_B) \cdot p(\beta_{B}|\sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu) \cdot \ldots \cdot p(\mu) \cdot p(\sigma^{-2}) \cdot p(\mathbf{x}_{\mathcal{L}}) \cdot p(\mathbf{\lambda}_{\mathcal{L}})$$

(7)

A graphical model representation of the new regression model is provided in Figure 1. The full conditional distributions of $\beta_B, \sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}$, and $\mu$ can be computed analytically, so that Gibbs-sampling can be applied to generate a posterior sample. As the derivations are mathematically involved, we relegate them to Part A of the Supplementary Material.

The marginalization rule from Section 2.3.7 of Bishop (2006) yields:

$$p(\mathbf{y}|\mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu) = \frac{\Gamma \left( \frac{n_{0}+a}{2} \right)}{\Gamma(a)} \cdot \frac{\Gamma \left( \frac{n_{0}+b}{2} \right)}{|C|^n_{2}}$$

$$\ldots \cdot \frac{\Gamma \left( \frac{n_{0}+b}{2} \right)}{|C|^n_{2}}$$

$$\ldots \cdot \left( 2b + (y - X_B \mu)^T C^{-1} (y - X_B \mu) \right)^{\frac{n_{0}+b}{2}}$$

where $T := \sum_{k=1}^{K} (T_k - 1)$, and $C := 1 + X_B \Sigma X_B^T$.

2.2 Inferring the relevant covariates and their types

In typical applications, there is a set of $N$ variables, and the subset of the relevant covariates has to be inferred from the data. Each covariate can be either constant ($\delta = 1$) or component-specific ($\delta = 0$). Let $\Pi = \{X_1, \ldots, X_K\}$ be a subset of the $N$ variables, and let $\delta = (\delta_0, \delta_1, \ldots, \delta_n)^T$ be a vector of binary variables, where $\delta_i$ indicates whether $X_i$ has a constant ($\delta_i = 1$) or component-specific ($\delta_i = 0$) regression coefficient. The first element, $\delta_0$, refers to the intercept.

The goal is then to infer the covariate set $\Pi$ and the corresponding indicator vector $\delta$ from the data. For any combination of $\Pi$ and $\delta$, the partitioned design matrix $X_B = X_{\Pi \Pi a}$ can be built, and the marginal likelihood $p(\mathbf{y}|\mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu, \Pi, \delta)$ can be computed with (8). We get for the posterior:

$$p(\Pi, \delta, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}, \mu, \Pi, \delta) \cdot p(\Pi) \cdot p(\delta) \cdot \ldots \cdot p(\mu) \cdot p(\delta) \cdot p(\mathbf{x}_{\mathcal{L}}) \cdot p(\mathbf{\lambda}_{\mathcal{L}})$$

where $p(\mu, \Pi, \delta)$ is a Gaussian, whose dimension is the number of component-specific coefficients. For the covariate sets, $\Pi$, we follow Grzegorczyk and Husmeier (2013) and assume a uniform distribution, truncated to $|\Pi| \leq 3$. The prior $p(\delta)$ will be specified in Section 2.5. To generate samples from the posterior, we use a Markov Chain Monte Carlo (MCMC) algorithm, which combines the Gibbs-sampling steps for $\beta_B, \sigma^2, \mathbf{x}_{\mathcal{L}}, \mathbf{\lambda}_{\mathcal{L}}$, and $\mu$ with two blocked Metropolis Hastings (MHs) moves. In the first MH move the vector $\delta$ is sampled jointly with $\mu$, and in the second MH move $\Pi$ is sampled jointly with $\delta$ and $\mu$. As the implementation of the MCMC algorithm is involved, we relegate the mathematical details to Parts B and C of the Supplementary Material.

2.3 Competing models

A homogeneous model merges all data, while a non-homogeneous model assumes each component $k$ to have specific parameters; see (2). The new partially non-homogeneous model infers the best trade-off: Each regression coefficient can be either constant or component-specific.

For a fair comparison, we also allow the non-homogeneous model to switch between a homogeneous and a non-homogeneous
state. However, like all models that have been proposed so far, it operates on the covariate sets. All covariates have either component-specific \((S = 0)\) or constant \((S = 1)\) regression coefficients. In our method comparison, we include:

- DBN: A homogeneous model that merges all data, see (3).
- NH-DBN: The NH-DBN model switches between two states. We have a DBN for \(S = 1\), and the likelihood takes the form of (2) for \(S = 0\).
- coupled NH-DBN: This model from Grzegorzyk and Husmeier (2013) is an NH-DBN that globally couples the regression coefficients.

### 2.4 Specifying the covariate type prior

The NH-DBNs can switch between: 'all covariates are constant' \((S = 1)\) versus 'all covariates are component-specific' \((S = 0)\). Those states refer to \(\delta = 1\) and \(\delta = 0\) of the partially NH-DBN. To match the priors, we set:

\[
p(S = 1) = \frac{p(\delta = 1)}{p(S = 0)} = \frac{p(\delta = 1 | \mathbf{I})}{p(\delta = 0 | \mathbf{I})} \tag{9}
\]

For \(\mathbf{I} = \{X_1, \ldots, X_n\}\), \(\delta\) contains \(n + 1\) binary elements, which we assume to be independently Bernoulli distributed. To fulfil (9) the Bernoulli parameter must depend on \(n = |\mathbf{I}|\). We get:

\[
p(\delta = 1 | \mathbf{I}) = 0^\alpha n + 1 \quad \text{and} \quad p(\delta = 0 | \mathbf{I}) = (1 - \theta_n)^{n+1}.
\]

From (9) we obtain:

\[
r := \frac{p(S = 1)}{p(S = 0)} = \frac{\theta_n^{(n+1)}}{(1 - \theta_n)^{n+1}} \Rightarrow \theta_n = \left(\frac{r}{1 + r}\right)^{1/(n+1)}
\]

and

\[
p(\delta | \mathbf{I}) = \sum_{\theta_n} b^n \theta_n b^{(1 - \theta_n)}
\]

For mixture models it is often assumed that the number of components \(K\) has a Poisson distribution (Green, 1995). We truncate it to \(K \in \{1, K\}\):

\[
p(S = 0) = \frac{q(K)}{q(1) + q(K)}, \quad p(S = 1) = \frac{q(1)}{q(1) + q(K)}
\]

where \(q(.)\) is the density of the Poisson distribution with parameter \(\theta = 1\).

### 2.5 GP smoothing for non-equidistant data

The regression models assume that the time lag \(\mathcal{O}\) between the response value \(y_{k,t_i}\) and the covariate values \(x_{k,t_1}, \ldots, x_{k,t_i}\) is the same for all \(t_i\). If the data within a component \(k\) were measured at time points \(t_1, \ldots, t_{T_k}\) with varying distances \(\mathcal{O}_i := t_i - t_{i-1}\), the models lead to biased results. For this scenario, we propose to replace the observed non-equidistant response values by predicted equidistant response values. We propose the following GP-based method:

- Determine the lowest time lag \(\mathcal{O} = \min\{\mathcal{O}_2, \ldots, \mathcal{O}_{T_k}\}\), where \(\mathcal{O}_i := t_i - t_{i-1}\).
- Given the observed data points \(\{(t, y_{k,t_i}) : t = t_1, \ldots, t_{T_k}\}\), use a GP to predict the whole curve \(\{(t, y_{k,t})\}_{t \in \mathcal{O}}\).
- Extract the response values at the time points:
  \(t_i + \mathcal{O}, \ldots, T_k + \mathcal{O}\).
- Build the response vector and design matrix such that the values \(x_{i,k,1}, \ldots, x_{i,k,T_k}\) are used to explain the predicted response value \(y_{i,k,T_k + \mathcal{O}}\) (\(i = 1, \ldots, T_k\)). The new lag is then constant \(\mathcal{O}_i = \mathcal{O}\).

A GP is a stochastic process \(\{Y_k\}_{t \in \mathcal{O}}\) here indexed by time, such that every finite subset of the random variables has a Gaussian distribution. A GP can be used to estimate a non-linear curve (\(y_k(t_i) \in \mathcal{O}\) from noisy observations. We here assume the relationship:

\[
y_{k,t} = f(t) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)
\]

is observational noise, and the non-linear function \(f(.)\) is unknown. We estimate \(f(.)\) by fitting a GP to the observed data. The GP defines a distribution over the functions \(f(.)\), which transforms the input \((t_1, \ldots, t_{T_k})\) into output \((y_{k,t_1}, \ldots, y_{k,t_{T_k}})\), such that

\[
(Y_{k,1}, \ldots, Y_{k,T_k})^T \sim \mathcal{N}(0, \mathbf{K} + \sigma^2 \mathbf{I}) \tag{10}
\]

where \(\mathbf{I}\) is the identity matrix, and the elements of the \(T_k\)-by-\(T_k\) covariance matrix, \(\mathbf{K}\), are defined through a kernel function: \(K_{ij} = \xi^2 \cdot k(t_i, t_j)\) with signal variance parameter \(\xi^2\). The kernel function \(k(.)\) is typically chosen such that similar inputs \(t_i\) and \(t_j\) yield correlated variables \(Y_{k,t_i}\) and \(Y_{k,t_j}\). A popular and widely used kernel is the squared exponential kernel with:

\[
k(t_i, t_j) = \exp\left(-\frac{(t_i - t_j)^2}{2l^2}\right)
\]

where \(l\) is the length scale. For the observed vector \(y_{k,t} := (y_{k,t_1}, \ldots, y_{k,t_{T_k}})^T\) we then have the predictive distribution:

\[
y_{k,t} \sim \mathcal{N}(\hat{y}_{k,t}, \Sigma_{k,t}) \tag{11}
\]

with

\[
\hat{y}_{k,t} := (K^* + \sigma^2 \mathbf{I})^{-1} \cdot (K^* + \sigma^2 \mathbf{I})^{1/2} \cdot y
\]

\[
\Sigma_{k,t} := (K^* + \sigma^2 \mathbf{I})^{-1} - (K^* + \sigma^2 \mathbf{I})^{-1} (K^* + \sigma^2 \mathbf{I})^{1/2} \tag{12}
\]

where \(y := (y_{k,1}, \ldots, y_{k,T_k})^T\) is the observed response vector, and \(K^*\) and \(\mathbf{K}\) are \(T_k\)-by-\(T_k\) matrices, whose elements are given by:

\[
K_{ij} := \xi^2 \cdot k(t_i + \mathcal{O}, t_j + \mathcal{O})\quad \text{and} \quad (K^*')_{ij} := \xi^2 \cdot k(t_i + \mathcal{O}, t_j + \mathcal{O}).
\]

Before inferring the GP, we standardize \(y\) to mean 0, and we impose log-uniform priors on the GP parameters \((\sigma^2, \xi^2\) and \(l\). For predicting the unobserved response vector, we have to make two decisions:

1. The GP parameters can either be sampled via MCMC simulations or their maximum a posteriori (MAP) estimates can be computed.
2. Given GP parameters, the vector \(y_{k,t}\) can be sampled from (11) or it can be set equal to its predictive expectation, \(\hat{y}_{k,t}\), defined in (12).

We have implemented and cross-compared all four combinations. For lack of space, we here report the results obtained for predictive expectations based on MAP estimates. A comparison of the four approaches can be found in Part D of the Supplementary Material.

### 2.6 Learning topologies of regulatory networks

Assume that the variables \(Z_1, \ldots, Z_N\) interact with each other in a form of a network and that data were collected under \(K\) conditions and that the conditions influence some of the interactions. Let \(\mathbf{D}_k\) denote the \(N\)-by-\(T_k\) data matrix which was measured under condition \(k\). The rows of \(\mathbf{D}_k\) correspond to the variables and the columns of \(\mathbf{D}_k\) correspond to \(T_k\) time points. \(\mathbf{D}_{k,t}\) denotes the value of \(Z_t\) at time point \(t\) under condition \(k\).

The goal is to infer the network structure. Interactions for temporal data are usually modelled with a time lag, e.g. of order \(\mathcal{O} = 1\). An edge, \(Z_i \rightarrow Z_j\), indicates that \(Z_i\) has an effect on \(Z_j\) in the following sense: For all \(k\) the value \(\mathbf{D}_{k,t+1}(Z_i, t)\) depends on \(\mathbf{D}_{k,t}(Z_i, t)\).

There is noacyclicity constraint, and DBN inference can be thought of as inferring \(N\) separate regression models and combining the results. In the ith model \(Y_i = Z_i\) is the response. The remaining \(N_i := N - 1\) variables \(Z_1, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_N\) are the potential
covariates. For each \( Y := Z_r \) we infer a covariate set \( \Pi_r \), and the covariate sets \( \Pi_1, \ldots, \Pi_r \) describe a network \( \mathcal{N} \). There is the edge \( Z_r \rightarrow Z_i \) in the network \( \mathcal{N} \) if and only if \( Z_i \in \Pi_r \).

We can thus apply the partially non-homogeneous model to each \( Y = Z_r \) separately, to generate posterior samples. We extract the covariate sets, \( \Pi_1^{(1)}, \ldots, \Pi_r^{(R)} \) (\( i = 1, \ldots, N \)), and we merge them to a network sample \( \mathcal{N}^{(1)}, \ldots, \mathcal{N}^{(R)} \). The \( r \)th network \( \mathcal{N}^{(r)} \) possesses the edge \( Z_r \rightarrow Z_i \) if and only if \( Z_i \in \Pi_r^{(r)} \). For each edge \( Z_r \rightarrow Z_i \) we can then estimate its marginal posterior probability (‘score’):

\[
\hat{s}_{ij} = \frac{1}{R} \sum_{r=1}^{R} I_{i \rightarrow j}(\mathcal{N}^{(r)}) \quad \text{where} \quad I_{i \rightarrow j}(\mathcal{N}^{(r)}) = \begin{cases} 1 & \text{if } Z_i \in \Pi_j^{(r)} \\ 0 & \text{if } Z_i \notin \Pi_j^{(r)} \end{cases}
\]

When the true network is known, we can evaluate the network reconstruction accuracy using precision-recall curves. For each \( \psi \in [0, 1] \) we extract the \( n(\psi) \) edges whose scores \( \hat{s}_{ij} \) exceed \( \psi \), and we count the number of true positives \( T(\psi) \) among them. Plotting the precisions \( P(\psi) := T(\psi)/n(\psi) \) against the recalls \( R(\psi) := T(\psi)/M \), where \( M \) is the number of edges in the true network, gives the precision-recall curve (Davis and Goadrich, 2006). We refer to the area under the curve as AUC value. The higher the AUC, the higher the reconstruction accuracy.

### 3 Implementation

For the inverse Gamma distributed parameters \((\sigma^2, \lambda_1^2, \lambda_2^2)\) we use shape and rate parameters from earlier works, e.g. in Grzegorczyk and Husmeier (2013) and Lèbre et al. (2010): \( \sigma^{-2} \sim \text{Gam}(0.005, 0.005) \) and \( \lambda_1^{-2}, \lambda_2^{-2} \sim \text{Gam}(2, 0.2) \) and for the hyperprior on \( \mu \) we use \( \nu_1 = 0 \) and \( \Sigma_0 = I \). Other settings led to comparable results what indicates robustness w.r.t. those hyperparameters. To ensure a fair comparison we use the same hyperparameters for the competing models; cf. Section 2.3.

For generating posterior samples, we run the MCMC algorithm from Section 2.2 for 100,000 (100k) iterations. We set the burn-in phase to 50k and we sample every 100th graph during the sampling phase. This yields \( R = 500 \) posterior samples for each response \( Y = Z_r \). We merge the individual covariate sets \( \Pi_i^{(r)} \) (\( i = 1, \ldots, N; r = 1, \ldots, R \)) to a network sample \( \mathcal{N}^{(1)}, \ldots, \mathcal{N}^{(R)} \), as explained in Section 2.6. For each edge \( Z_r \rightarrow Z_i \) we then compute its edge score \( \hat{s}_{ij} \).

We used scatter plots of edge scores from independent simulations to monitor convergence. In Section 4.3 we study convergence, scalability and the computational costs for model inference.

We implement the GP method with the squared exponential kernel and used the Matlab package ‘GPstuff’ (Vanhatalo et al., 2013) to numerically determine the MAP estimates of the parameters via scaled conjugate gradient optimization. We also tested other kernels, such as the Matern 3/2 and 5/2 kernel, and for them we obtained very similar results.

### 4 Empirical results

#### 4.1 Pre-study 1: GP smoothing

Our first objective is to provide empirical evidence that the proposed GP method from Section 2.5 can yield substantial improvements. To this end, we generate values for 10 autoregressive (AR) variables:

\[
X_{ij} = \sqrt{1 - \eta} \cdot X_{i,j-1} + \sqrt{\eta} \cdot \epsilon_{ij} \quad (t = 0, \ldots, 120; i = 1, \ldots, 10)
\]

where all \( \epsilon_{ij} \)’s are independently \( N(0, 1) \) distributed, \( X_{1,t} \sim N(0, 1) \) for all \( i \) and \( \eta \in (0, 1) \). This yields \( X_{i,1} \sim N(0, 1) \) for all \( t \) and all \( i \). We further assume that \( X_1 \) and \( X_2 \) are covariates for:

\[
Y_{i,t+1} = \beta_0 + \beta_1 X_{i,t} + \beta_2 X_{i,t} + \epsilon_{i,t+1}
\]

where \( \epsilon_{i,t+1} \)’s are independently \( N(0, \eta^2) \).

In a second scenario we replace (13) by moving averages (MA):

\[
X_{i,t} = \sum_{j=t-g}^{t} \epsilon_{ij} \quad (t = 0, 1, \ldots, 120; i = 1, \ldots, 10)
\]

where all \( \epsilon_{ij} \)’s are independently \( N(0, (q + 1)^{-1}) \) distributed, so that again \( X_{i,t} \sim N(0, 1) \) for all \( i \) and \( t \).

We generate data for both scenarios (AR and MA) with different parameter settings \((\beta_0, \beta_1, \beta_2)\) in (14) and \( \eta \) in (13), respecting \( q \) in (15). We thin the data out and keep only the observations at the time points \( t \in \{0, 1, 3, 5, 10, 15, 30, 45, 60, 120\} \), as the same time points were measured for the mTORC1 data; see Section 4.6.

The standard regression approach uses the covariate values at \( t_i \) for explaining \( Y \) at \( t_{i+1} \), although the time lag steadily increases. The GP method from Section 2.5 predicts the response values at \( t_{i+1} \) and replaces \( y_{i+1} \) (observed \( Y \) at \( t_{i+1} \)) by \( \hat{y}_{i+1} \) (predicted \( Y \) at \( t_{i+1} \)), where \( \hat{Y} = 1 \). With both approaches we run MCMC simulations on each dataset, and from the MCMC samples we compute for each covariate \( X_i \) the score that \( X_i \) is a covariate for \( Y \). Our results show that the proposed GP method finds the true covariates \( X_1 \) and \( X_2 \), while the standard approach cannot clearly distinguish them from the irrelevant variables \( X_3, \ldots, X_8 \). Figure 2 shows histograms of the average covariate scores for AR data with \( \beta_1 = 1 \) and \( \eta = 0.2 \), and for MA data with \( \beta_1 = 1 \) and \( q = 10 \).

#### 4.2 Pre-study 2: Synthetic RAF-pathway data

The RAF pathway, as reported in Sachs et al. (2005), consists of \( N = 11 \) nodes and 20 directed edges. The topology of the RAF pathway is shown in Part E of the Supplementary Material. We generate data with \( K = 2 \) components and \( T_k = 10 \) data points each.
The parent nodes of each node \( Z_i \) build its covariate set \( \Pi_i \). We assume a linear model with component-specific regression coefficients:

\[
\tilde{z}_{ik,t+1} = \beta'_{k,0} + \sum_{j=1}^{k-1} \beta_{k,j} \cdot \tilde{z}_{ik,t} + \epsilon_{ik,t} \quad (k = 1, 2)
\]

where \( \tilde{z}_{ik,t} \) denotes the value of node \( Z_i \), at time point \( t \) in component \( k \), and \( \beta_{k,j} \) is the regression coefficient for \( Z_{ij} \) \( \rightarrow \) \( Z_{ik} \) in component \( k \). The noise values \( \epsilon_{ik,t} \) and the initial values \( \tilde{z}_{ik,1} \) are sampled from independent \( N(0, 0.05^2) \) distributions. For \( Z_i \), there are \( 2(\Pi_i + 1) \) component-specific regression coefficients. For each \( Z_i \), we collect them in two vectors \( \beta_{k,i} \) (\( k = 1, 2 \)), and we sample the elements of \( \beta_{k,i} \) from \( N(0, 1) \) Gaussian distributions. We then re-normalize the vectors to Euclidean norm one: \( \beta_{k,i} \leftarrow \beta_{k,i} / ||\beta_{k,i}|| \) (\( k = 1, 2 \)). We distinguish six scenarios:

- (S1) Identical: We withdraw \( \beta_{k,i} \) and assume that the same regression coefficients apply to both components. We set: \( \beta_{k,i} = \beta_{j,i} \) for all \( k \).
- (S2) Identical signs (correlated): We enforce the coefficients to have the same signs, i.e. we replace \( \beta_{k,i,j} \) by: \( \beta_{k,i,j} := \text{sign}(\beta_{j,i}) \cdot ||\beta_{j,i}|| \) for all \( i \) and \( j \).
- (S3) Uncorrelated: We use the vectors \( \beta_{k,i} \) for component \( k \) (\( k = 1, 2 \)). The component-specific coefficients \( \beta_{k,i,j} \) and \( \beta_{j,i,j} \) are then uncorrelated for all \( i \) and \( j \).
- (S4) Opposite signs (negatively correlated): We withdraw the vector \( \beta_{k,i} \) and we set: \( \beta_{k,i,j} = (-1) \cdot \beta_{j,i} \). The coefficients \( \beta_{k,i,j} \) and \( \beta_{j,i,j} \) are then negatively correlated.
- Mixture of (S1) and (S3): We assume that 50% of the coefficients are identical for both \( k \), while the other 50% are uncorrelated. We randomly select 50% of the coefficients and set: \( \beta_{k,i,j} = \beta_{j,i,j} \). The other 50% of the coefficients stay unchanged (uncorrelated).
- Mixture of (S1) and (S4): We withdraw \( \beta_{k,i} \) and we assume that 50% of the coefficients are identical for both \( k \), while the other 50% have an opposite sign. We randomly select 50% of the coefficients and set: \( \beta_{k,i,j} = (-1) \cdot \beta_{j,i,j} \). For the other coefficients we set \( \beta_{k,i,j} = \beta_{j,i,j} \).

For each scenario we generate 25 datasets. We then analyse every dataset with each model. Figure 3 shows the average AUC values for reconstructing the RAF pathway. Only for scenario (S1), where all coefficients are constant, the models perform equally well. For (S2–S6) the homogeneous DBN is substantially worse than the NH-DBNs. The coupled NH-DBN is slightly superior to the (non-coupled) NH-DBN. The proposed partially NH-DBN yields the highest average AUC scores.

4.3 Pre-study 3: Scalability and computational costs

To study the scalability of the new network reconstruction method, we generate data for random network structures with \( N \in \{10, 25, 50, 100\} \) nodes. For each node \( Z_i \), we first sample the number of parents \( x_i \) from a Poisson distribution with parameter \( \lambda = 1 \) (‘Poisson in-degree distribution’), before we randomly draw a parent set \( \Pi_i \) from a uniform distribution over the system of all parent sets with cardinality \( x_i \) \( \{\Pi_i : ||\Pi_i|| = x_i\} \). Given the network structure, we generate data as described in Section 4.2, i.e. via regression relationships using \( K = 2 \) and \( T_k = 10 \). Here we present and discuss the results for the scenarios: ‘Mixture of (S1) and (S3)’. For each \( N \) we generate 10 independent datasets, i.e. 40 in total. Next, we measure how many MCMC iterations \( W \) we can perform in 1 h. With our Matlab implementation on a desktop computer with Intel Xeon 2.5 GHz processor and 8GB of RAM, the average numbers of iterations per hour are: \( W = 208.637 \) \((N = 10)\), \( W = 83.615 \) \((N = 25)\), \( W = 41.666 \) \((N = 50)\) and \( W = 19.855 \) \((N = 100)\).

To monitor convergence and network reconstruction accuracy in real-time, we perform long MCMC simulations. For each \( N \) we select the numbers of iterations such that the simulation takes 16 h. During the simulations we sample 200 equidistant networks per hour (i.e. 3200 networks in total). When withdrawing the first 50% of the networks (‘burn-in’), we have \( R_0 = 100b \) networks after \( h \) hours. We use those \( R_0 \) networks to assess the performance after \( h \) hours of computational time. For running two independent 16 h long MCMC simulations on each of the 40 datasets (computational time: 1280 h), we use a computer cluster.

To assess convergence, we consider scatter plots of the edge scores of two independent MCMC simulations on the same dataset. For the largest networks with \( N = 100 \), Figure 4a shows superimposed scatter plots for different computational times. It can be seen that after 2–4 h only few edge points deviate from the diagonal, i.e. only few edge scores differ between independent simulations. This is a good indication of convergence. The corresponding scatter plots for the smaller networks with \( N \in \{10, 25, 50\} \) can be found in Part F of the Supplementary Material. As expected, the Supplementary Figures show that the rate of convergence decreases with the size of the network \( N \). The upper panel of Figure 4b monitors the average precision-recall AUC scores along the computational time. The four AUC curves run into plateaus. Already after 1–2 h the AUCs (i.e. the average network reconstruction accuracy) do not improve anymore. The two curves for \( N = 10 \) and 25 converge to AUC = 0.72, while the AUC limits are lower for \( N = 50 \) (AUC = 0.58) and \( N = 100 \) (AUC = 0.49). The individual AUC curves are shown in Part F of the Supplementary Material. Taking into account that a network among \( N = 100 \) nodes had to be inferred from 20 data points (\( K = 2 \) conditions with \( T_k = 10 \) data points each), the rather low network reconstruction accuracy (AUC = 0.49) is not surprising. To show that higher AUCs can be reached for networks with \( N = 100 \) nodes, we repeat the study with \( T_k = 20 \) and \( 40 \). The bottom panel of Figure 4b monitors the average AUC scores for \( N = 100 \) and \( T_k \in \{10, 20, 40\} \). Here, we had to adapt the numbers of MCMC iterations per hour to \( W = 12707 \) \((T_k = 20)\), and \( W = 9343 \)
The new curves also run into plateaus and reach higher limits: $\text{AUC} = 0.70$ and $\text{AUC} = 0.83$. The results of this section are compactly summarized in Table 1.

### 4.4 Reconstructing the yeast gene network topology

By means of synthetic biology, Cantone et al. (2009) designed a network with $N = 5$ genes in *S.cerevisiae* (yeast); Figure 5 shows the true network. With quantitative Real-Time Polymerase Chain Reaction, Cantone et al. (2009) then measured in vivo gene expression data: under galactose- ($k = 1$) and glucose-metabolism ($k = 2$). $T_1 = 16$ measurements were taken in galactose and $T_2 = 21$ in glucose. The data have become a benchmark application, as the network reconstruction accuracies can be cross-compared on real in vivo gene expression data. Figure 5 shows the results, and again a clear trend can be seen: The homogeneous DBN yields the lowest AUC value. The NH-DBN model yields higher AUCs and can be further improved by coupling the regression coefficients (coupled NH-DBN). The proposed partially NH-DBN reaches the highest network reconstruction accuracy. The results are thus consistent with the results for the RAF-pathway data in Section 4.2.

### 4.5 The circadian clock network in *A.thaliana*

The circadian clock network in *A.thaliana* orchestrates the gene regulatory processes, related to the plant metabolism, with respect to the daily changing dark: light cycles of the solar day. The mechanism of internal time-keeping allows the plant to anticipate each new day, at the molecular level, and to optimize its growth. In $K = 4$ experiments *Arabidopsis* plants were entrained in different dark: light cycles, before the gene expressions of $N = 9$ circadian clock genes were measured under experimentally controlled constant light condition. The numbers of observed time points are $T_1 = 12$ and $T_k = 13$ for $k = 2, 3, 4$. For further details on the experimental protocols we refer to Grzegorczyk (2016). Figure 6a shows a network that was inferred with the new partially coupled model. For the prediction, we extracted the 21 edges with edge scores higher than the threshold $\psi = 0.5$. A proper biological evaluation of the network is hindered and beyond the scope of this article, as the true circadian clock network has not been fully discovered yet. However, our predicted network in Figure 6a contains many edges that are consistent with hypotheses from the plant biology literature. In particular, the high-scoring feedback loop LHY $\rightarrow$ TOC1 seems to be the most important key feature of the circadian clock network (see, e.g. Locke et al., 2006). Moreover, it has been reported that LHY is a regulator of the genes ELF3 and ELF4 (Alabadi et al., 2001; Kikis et al., 2005). Also the edge LHY $\rightarrow$ CCA1 is not unexpected, as LHY and CCA1 are known to be biological homologues (Miwa et al., 2007).
Four more edges, for which we could find biological literature references, are: GI $\rightarrow$ TOC1 (Locke et al., 2005), ELF3 $\rightarrow$ LHY (Kikis et al., 2005), ELF3 $\rightarrow$ TOC1 (Dixon et al., 2011) and ELF3 $\rightarrow$ PRR9 (Chow et al., 2012).

### 4.6 The mTORC1 network

The mammalian target of rapamycin complex 1 (mTORC1) is a serine/threonine kinase which is evolutionary conserved and essential in all eukaryotes (Saxton and Sabatini, 2017). mTORC1 is at the centre of a multiply wired, complex signalling network, whose topology is well studied and contains several well-characterized feedback loops (Saxton and Sabatini, 2017). Hence, we used the mTORC1 network as a surrogate based on which we can objectively evaluate the predictive power of our partially NH-DBN model for learning network structures. The signalling network converging on mTORC1 is built by kinases, which inactivate or activate each other by phosphorylation. Thus, a protein can be phosphorylated at one or several sites, and the phosphorylations at these positions determine its activity. Signalling through the mTORC1 network is elicited by external signals like insulin or amino acids. Dalle Pezze et al. (2016) relatively quantified 11 phosphorylation states of 8 key proteins across the mTORC1 signalling network by immunoblotting; for an overview see Table 2. Dynamic time course data were obtained under two experimental conditions, namely upon stimulation with amino acids only ($k = 1$), and with amino acids plus insulin ($k = 2$). The phosphorylation states were measured at $T_k = 10$ time points: $t = 0, 1, 3, 5, 10, 15, 30, 45, 60, 120$ minutes, so that the time lag increases from 1 to 60. We therefore apply the GP method from Section 2.5 to predict equidistant response values, before analysing the data with the proposed partially NH-DBN. The 12 edges with scores higher than $\psi = 0.5$ yield the network topology shown in Figure 6b. A literature review shows that 11 out of the 12 edges have been reported earlier.

We focus first on the five interactions with the highest scores $\psi > 0.8$. Two out of these five interactions are enzyme-substrate relationships: p70-S6K is a kinase which is directly activated by mTORC1 through phosphorylation at threonine 389 [p70-S6K-pT389] (Saxton and Sabatini, 2017). Thus, p70-S6K-pT389 represents a direct readout of mTORC1 activity. p70-S6K phosphorylates IRS1 at serine 636, [IRS1-pS636] (Tsatsos and Kandor, 2006) and mTOR at serine 2448 [mTOR-pS2448] (Dibble and Cantley, 2015), and both edges are correctly identified by our model [p70-S6K-pT389 $\rightarrow$ IRS1-pS636, p70-S6K-pT389 $\rightarrow$ mTOR-pS2448]. Two other interactions with a high score are between AKT-pT308 $\rightarrow$ AKT-pS473. The two phosphorylations are predicted by our model to influence each other, and a positive feedback between phosphorylation events on S473 and T308 of AKT has indeed been demonstrated biochemically (Manning and Toker, 2017). Another high score prediction is between IRS1-pS636 and p70-S6K-pT389 (Tsatsos and Kandor, 2006). Phosphorylation at S636 inhibits IRS1, thereby leading to inhibition of mTORC1 and its substrate p70-S6K-T389 (Tsatsos and Kandor, 2006). Thus, the negative feedback between IRS1-pS636 and p70-S6K-pT389 explains the learned edge between them [IRS1-pS636 $\rightarrow$ p70-S6K-pT389]. In addition, IRS1 inhibition by phosphorylation at S636 results in reduced

---

**Table 2. mTORC1 timecourse data**

<table>
<thead>
<tr>
<th>Protein</th>
<th>Full name</th>
<th>Sites</th>
</tr>
</thead>
<tbody>
<tr>
<td>mTOR</td>
<td>mammalian target of rapamycin</td>
<td>pS2481, pS2448</td>
</tr>
<tr>
<td>PRAS40</td>
<td>proline-rich AKT/PKB substrate 40 kDa</td>
<td>pT246, pS183</td>
</tr>
<tr>
<td>AKT</td>
<td>Protein kinase B</td>
<td>pT308, pS473</td>
</tr>
<tr>
<td>IRS1</td>
<td>insulin receptor substrate 1</td>
<td>pS636</td>
</tr>
<tr>
<td>IR-beta</td>
<td>insulin receptor beta</td>
<td>pY1146</td>
</tr>
<tr>
<td>AMPK</td>
<td>AMP-dependent protein kinase</td>
<td>pT172</td>
</tr>
<tr>
<td>TSC2</td>
<td>tuberous sclerosis 2 protein</td>
<td>pS1387</td>
</tr>
<tr>
<td>p70-S6K</td>
<td>Ribosomal protein S6 kinase beta-1</td>
<td>pT389</td>
</tr>
</tbody>
</table>

*Note:* Overview to the 8 proteins and the 11 measured phosphorylation sites.
phosphorylation of AKT at threonine 308, which is in agreement with the learned edge between IRS1-pS636 and AKT-pT308 [IRS1-pS636—AKT-pT308].

We could also find evidence for 6 of the remaining 7 edges with scores in between 0.5 and 0.8. PRAS40 is an endogenous mTORC1 inhibitor (Saxton and Sabatini, 2017). The edge from PRAS40-pT246 to PRAS40-pS183 corresponds to a well-described mechanism of PRAS40 regulation: AKT phosphorylates PRAS40 at T246 [PRAS40-pT246], which allows subsequent phosphorylation of PRAS40-S183 by mTORC1 (Nascimento et al., 2010). This interaction is accurately resembled by our model [PRAS40-pT246—PRAS40-pS183]. PRAS40's double phosphorylation dissociates PRAS40 from mTORC1, leading to its derepression (Nascimento et al., 2010). This mechanism is resembled by the edge between PRAS40-S183 and mTOR-S2481 [PRAS40-pS183—mTOR-pS2481], the latter being an autophosphorylation site which directly monitors mTOR activity (Soliman et al., 2010). Furthermore, the model suggests an edge between p70-S6K-pT389 and PRAS40-pS183 [p70-S6K-pT389—PRAS40-pS183]. Both are mTORC1 substrate sites (Nascimento et al., 2010; Saxton and Sabatini, 2017) and are therefore often targeted in parallel. The only predicted edge for which there is to the best of our knowledge no literature evidence is between mTOR-pS2448 and TSC2-pS1387 [mTOR-pS2448—TSC2-pS1387]. TSC2 is activated by phosphorylation at S1387 and inhibits mTORC1 (Hindupur et al., 2015). Our model prediction that mTORC1—when phosphorylated at S2448 by p70-S6K—regulates TSC2 remains to be experimentally tested.

After having identified 11 of 12 edges as true positives, we performed a literature review to find false negative edges, i.e. edges that our model did not extract, although it has been reported that they exist. This way, we could identify two false negative edges, namely: IR-beta-pY1146—AKT-pT308 and AMPK-pT172—TSC2-pS1387, which were reported in Vigneri et al. (2016) and Mihaylova and Shaw (2012), respectively. Finally, we note that this does not imply that the remaining edges that our model did not extract can be assumed to be true negatives. Nowadays incomplete knowledge about the topology of the mTORC1 pathway renders the safe identification of true negative edges impossible. The absence of literature reports on an edge does not necessarily imply that it does not exist.

5 Conclusion and discussion

We propose a new partially NH-DBN model for learning network structures. When data are measured under different experimental conditions, it is rarely clear whether the data can be merged and analysed within one single model, or whether there is need for a NH-DBN model that allows the network parameters to depend on the condition. The new partially NH-DBN has been designed such that it can infer the best trade-off from the data. It infers for each individual edge whether the corresponding interaction parameter is constant or condition-specific. Our applications to synthetic RAF pathway data as well as to yeast gene-expression data have shown that the partially NH-DBN model improves the network reconstruction accuracy. We have used the partially NH-DBN model to predict the structure of the mTORC1 signalling network. As the measured mTORC1 data are non-equidistant, we have applied a GP-based method to predict the missing equidistant values. Results on synthetic data (see Section 4.1) show that the proposed GP-method (see Section 2.5) can lead to substantially improved results.

All but one of the predicted interactions across the mTORC1 network are reflected in experiments reported in the biological literature. Dalle Pezze et al. (2016) built an ODE-based dynamic model which allows to predict signalling responses to perturbations. Like for many ODE-based models, the topology of this model was defined by the authors, based on literature-knowledge. The ODE model simulations could reproduce the measured mTORC1 time-course data. Interestingly, all the connections predicted by our new partially NH-DBN model form part of the core model by Dalle Pezze et al. (2016). Hence, we present an alternative unsupervised learning approach, in which the topology of signalling networks is inferred directly from the data. The new model is thus a complementary tool that enhances dynamic model building by predicting the network’s topology in a purely data-driven manner.

Although it worked well for the mTORC1 data, we note that the GP method from Section 2.5 requires the time series to be sufficiently smooth. For non-smooth time series the method might not be able to properly predict the values at unobserved time points, leading to biased response values. Then, network reconstruction methods, such as the new partially NH-DBN, will inevitably infer distorted network topologies and wrong conclusions might be drawn. The assumption of smoothness is therefore crucial for the complete analysis pipeline to work. Our results in Section 4.3 show that the new partially NH-DBN can also be used to infer larger networks. However, our results suggest that there is then need for more data points (or higher signal-to-noise ratios, respectively) to reach accurate network predictions. A conceptual advantage of our partially NH-DBN is that it has two established models, namely the homogeneous DBN (δ = 1) and the globally coupled NH-DBN (δ = 0) as limiting cases. The new model operates between them, and as we follow a model averaging approach, it is less susceptible to over-fitting. The edge scores of the partially NH-DBN take the two established models as well as all ‘in-between’ models into account. For sparse data, we would thus expect low edge scores, indicating that we might average across too many models.

Funding

M.S.K. and M.G. are supported by the European Cooperation in Science and Technology (COST) [COST Action CA11109 European Cooperation for Statistics of Network Data Science (COSTNET)]. K.T. acknowledges support from the BBMB e; Med initiatives GloPATH [01ZX1402B] and MAPToR-NET (031A426B), the German Research Foundation [TH 1135/3-1], the Stichting TSC Fonds (calls 2015 and 2017) and the MESI-STRAT project, which has received funding from the European Union’s Horizon 2020 research and innovation programme under [grant agreement No 754688], K.T. is recipient of a Rosalind-Franklin-Fellowship of the University of Groningen and of the Research Award 2017 of the German Tuberous Sclerosis Foundation.

Conflict of Interest: none declared.

References

Kikis, E. et al. (2005) ELF4 is a phytochrome-regulated component of a negative-feedback loop involving the central oscillator components CCA1 and LHY. Plant J., 44, 300–313.