Constructing Bayesian Networks for Linear Dynamic Systems

Sander Evers
Radboud University Nijmegen
Inst. for Computing and Information Sciences
Nijmegen, the Netherlands

Peter J.F. Lucas
Radboud University Nijmegen
Inst. for Computing and Information Sciences
Nijmegen, the Netherlands

Abstract

Building probabilistic models for industrial applications cannot be done effectively without making use of knowledge engineering methods that are geared to the industrial setting. In this paper, we build on well-known modelling methods from linear dynamic system theory as commonly used by the engineering community to facilitate the systematic creation of probabilistic graphical models. In particular, we explore a direction of research where the parameters of a linear dynamic system are assumed to be uncertain. As a case study, the heating process of paper in a printer is modelled. Different options for the representation, inference and learning of such a system are discussed, and experimental results obtained by this approach are presented. We conclude that the methods developed in this paper offer an attractive foundation for a methodology for building industrial, probabilistic graphical models.

1 Introduction

As part of the manual construction process of a Bayesian network for an actual problem one needs to somehow translate knowledge that is available in the problem domain to an appropriate graphical structure. Problem characteristics also play a major role in the choice of the type of the associated local probability distributions. The whole process can be looked on as a form of *knowledge engineering*, where the actual construction of the Bayesian network is only one of the many needed activities in the development process. The acquisition of knowledge from domain experts is traditionally seen as one of the most important bottlenecks in knowledge engineering. It is well known that this can be greatly alleviated if there is an easy mapping from the informal ways domain knowledge is described, in documents or verbally by experts, to the Bayesian network formalism [7]. A typical example of such a mapping is the exploitation of available causal knowledge in a particular domain; often, causal knowledge can be easily translated to an initial graph structure of a Bayesian network, which can be refined later, for example by examining the conditional independence relationship represented by the resulting network. Fields where causal knowledge has been successfully used in knowledge engineering for Bayesian networks include medicine and biology.

However, for industrial applications the situation is somewhat different. This is mainly because in time, engineers have developed their own notational conventions and formalisms to get a grip on the domain of concern in the system-development process. In addition, industrial artifacts are designed by humans, and already early in the design process models are available that also can be used for other purposes: model-based design and development is here the central paradigm. Thus, rather than replacing methods from engineering by some new and unrelated methods, a better option seems to be to deploy as many of the engineering principles, methods, and assumptions as possible. Linearity is one of the assumptions frequently used, as it facilitates the development of complex models as industrial applications often are. Another commonly used method is the use of diagrams that act as abstractions of the system being developed. Diagrams act as important means for communication. Ideally, one would like to use similar diagrams as a starting point for building Bayesian networks or related probabilistic graphical models.

In this paper we explore these ideas by taking *Linear Dynamic Systems*, LDS for short, as a start for the construction process. LDS models enjoy a well-developed theory and practice, as they are widely used throughout many engineering disciplines for *tracking* system behaviour (cf. the well-known Kalman filter [4]) as well

as for *controlling* this behaviour. Like Bayesian networks, an LDS can often be represented by a graphical diagram, which facilitates documentation and communication of the model among experts and non-experts.

Although an LDS is deterministic in nature, it is often used in situations that involve uncertainty. The Kalman filter is the canonical example here: given some noisy observations, it determines the expected current state of the system (mean and variance). However, the Kalman filter only accounts for one specific type of uncertainty: additive linear Gaussian noise on the state and output variables.

In this article, we explore a different direction of augmenting an LDS with probabilities: we regard the *parameters* as unknown. This allows us for example to model a printer that heats different types of paper, in which it is uncertain what the current type is. Previous Bayesian networks modelling this situation were developed in a laborious ad hoc manner by close cooperation of domain experts and probabilistic modelling experts [3]; in this article we aim for a more systematic approach.

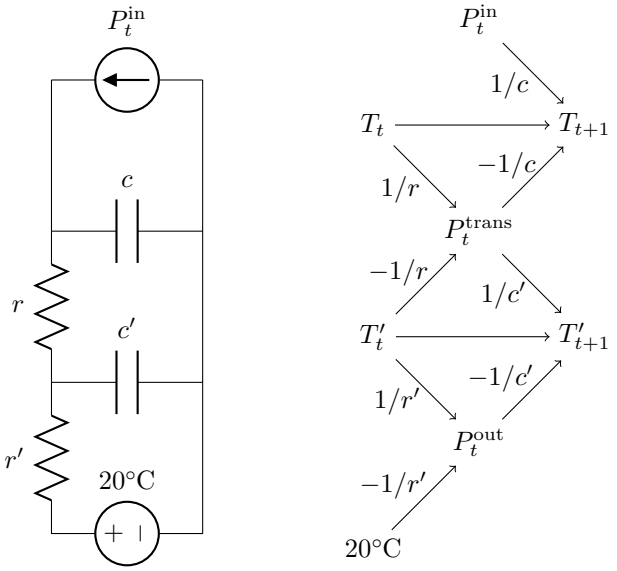

Figure 1: LDS model of paper heater. **Left:** as an electrical diagram, where voltage represents temperature and current represents heat flow. From top to bottom, the diagram consists of a (1) time-variable heat source, (2) heat mass with capacity c, (3) heat resistance r, (4) heat mass with capacity c', (5) heat resistance r', (6) heat sink with temperature 20°C modelling the constant flow of paper. **Right:** as a graphical representation of the state-space equations for two discrete time points $t, t{+}1$. The state of the system is described by the T variables, which represent the temperatures of the two heat masses. Its input is P_t^{in}, the power entering the system. Auxiliary variables represent the power flowing from the first heat mass to the second (P_t^{trans}), and from the second to the heat sink (P_t^{out}). **Note:** The *parameters* of the system are r, c, r', c' (instantiated with concrete values in a real system).

2 LDS models and their role in the engineering process

2.1 Basic definitions

In its most basic form, a *Linear Dynamic System* or LDS describes a system's behaviour by the differential equation

$$\tfrac{d}{dt}\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

known as the *state-space representation*. Here, vector $\mathbf{x}(t)$ represents the system's state at time t, $\mathbf{u}(t)$ its input at time t, and matrices $\mathbf{A}$ and $\mathbf{B}$ describe how the current *state change* depends linearly on the current state and input.

This is a continuous-time representation; in order to calculate with LDS models, time is usually discretized. In this article, we therefore use the simple discretized model

$$\mathbf{x}_{t+1} - \mathbf{x}_t = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t$$

in which the size of the discretization steps conveniently equals the unit of the time domain in order to simplify the exposition (in practice one can use discretization steps of size Δt and scale the $\mathbf{A}$ and $\mathbf{B}$ matrices with this factor). The equation can then be rewritten to:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{A}_d\mathbf{x}_t + \mathbf{B}_d\mathbf{u}_t \\ \mathbf{A}_d &= \mathbf{A} + \mathbf{I} \\ \mathbf{B}_d &= \mathbf{B} \end{aligned} \tag{1}$$

2.2 Role in engineering

In the engineering process, LDS models of systems are often represented by means of diagrams. We exemplify the role of these models and diagrams using a case study which remains our running example thoughout the paper. The case study originates from a manufacturer of industrial printers. To ensure print quality, paper needs to be heated to a certain temperature, which is accomplished by passing the paper along a metal heater plate. It is quite important that the paper reaches the right temperature. If it is too cold, print quality suffers; if it is too hot, energy (and money) is wasted or worse: the printer might malfunction. Therefore, engineers have put a lot of effort in the accurate modelling of the heating process. This results in models such as Fig. 1, in which the heater is modelled as two distinct heat masses: when the heater is powered, the first mass is directly heated, thereby indirectly heating the second mass, which transfers the

heat to the paper.[1] In the diagram, the heating dynamics are represented as an *electrical circuit*, where temperature plays the part of voltage and heat flow that of current. A diagram like this has important advantages for the engineering process:

- It is very well suited for *documentation and communication*. A trained engineer can read the system's basic dynamic behaviour off this diagram in a blink of an eye; for a non-expert with a science background it is relatively easy to gain some intuition.
- It has a *formal meaning* as an LDS; it translates into the state-space equations in the form of Eq. (1), connecting it to a vast body of theoretical and practical knowledge.
- It *separates qualitative and quantitative aspects* of the model; the former are determined by the diagram structure, the latter by the parameters.
- It is *composable*: other models like this can be developed independently and joined into a larger system.
- It is *supported by software*: drawing and managing modules of electrical circuits (and also other graphical forms like bond graphs [5] and schematic diagrams) can be done by tools like 20-sim [11], which can also perform the translation to the state-space representation. This representation can be used for simulation, e.g. in MATLAB.

However, it is confined to modelling deterministic behaviour. In the realm of probabilistic modelling, the formalism of *Bayesian networks* shares the above attractive properties. A natural question is therefore: how can we combine these well-known LDS models with Bayesian networks?

Specifically, this paper will explore the situation where the *parameters* of the system (in this case: r, c, r', c') involve uncertainty. This direction is induced by the following use case: the paper heater modelled above is used with different paper types $\{\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3\}$ (for example: 80 g/m^2 A4, 200 g/m^2 A4, 80 g/m^2 Letter). We have no direct knowledge about which type is in the heater, and would therefore like to model it as a probabilistic variable PT. Each paper type leads to a different value for the system's r' parameter (the heat resistance between plate and paper). The question we ask ourselves is: *How can we join the paper type variable to the LDS model, so we can infer probabilistically which paper type is in the heater, by observing the T' values of the system?*

[1] This is known as a *lumped element model*; in contrast, the heat distribution could also be modelled as a 3-dimensional temperature field over the plate.

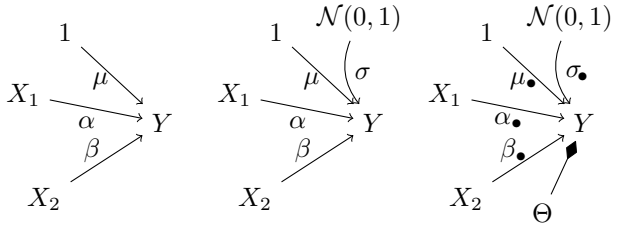

Figure 2: The three basic types of Bayesian network nodes we will use for LDS models. **Left:** Linear deterministic node $\mathsf{P}(y|x_1, x_2) = 1$ if $y = \mu + \alpha x_1 + \beta x_2$. **Middle:** Linear Gaussian node $\mathsf{P}(Y|x_1, x_2) = \mathcal{N}(\mu + \alpha x_1 + \beta x_2, \sigma^2)$. **Right:** Conditional linear Gaussian node $\mathsf{P}(Y|x_1, x_2, \theta) = \mathcal{N}(\mu_\theta + \alpha_\theta x_1 + \beta_\theta x_2, \sigma_\theta^2)$ (for discrete Θ). The notation is ours and is introduced in section 3.3.

3 Augmenting LDS models with uncertainty

3.1 Bayesian network representation

A *Bayesian network* $\mathcal{B} = (G, \Phi)$ is an acyclic directed graph G, consisting of *nodes* and *arcs*, that is faithful to a joint probability distribution factored as Φ, which contains for each node Y (with parents $X_1, X_2, \ldots$) a family of local conditional probability density or mass functions $\mathsf{P}(Y|X_1, X_2, \ldots)$. A *dynamic Bayesian network* [1] is a special case of this, where the nodes are partitioned in time slices all consisting of the same structure and distributions. Furthermore, arcs are only allowed between nodes in the same or adjacent time slice.

For modelling linear dynamic systems as (dynamic) Bayesian networks, only the following types of nodes are needed:

Deterministic nodes: a node Y with parents $X_1, X_2, \ldots$ is called *deterministic* if its conditional probability distribution is

$$\mathsf{P}(y|x_1, x_2, \ldots) = \begin{cases} 1 & \text{if } y = f(x_1, x_2, \ldots) \\ 0 & \text{if } y \neq f(x_1, x_2, \ldots) \end{cases}$$

for a certain function f; in this article, these functions are mostly *linear*, i.e.

$$y = \mu + \alpha x_1 + \beta x_2 + \ldots$$

We use a special notation for these *linear deterministic* nodes shown in Fig. 2 (left).

Linear Gaussians: a node Y with parents $X_1, X_2, \ldots$ is known in Bayesian network literature as a *linear Gaussian* if

$$\mathsf{P}(Y|x_1, x_2, \ldots) = \mathcal{N}(\mu + \alpha x_1 + \beta x_2 + \ldots, \sigma^2)$$

Networks that consist only of linear Gaussians (with $\sigma > 0$) have theoretical significance: their joint distribution is multivariate Gaussian, and exact inference is easy and efficient (e.g. see [6]). A linear Gaussian without parents $\mathcal{N}(\mu, \sigma^2)$ is simply called Gaussian; the Gaussian $\mathcal{N}(0, 1)$ is called a *standard* Gaussian. A linear Gaussian can be written as a linear deterministic node with two extra parents; see Fig. 2 (middle).

Conditional linear Gaussians: a node Y with parents $X_1, X_2, \ldots$ and discrete parent Θ is *conditional linear Gaussian* if

$$\mathsf{P}(Y|x_1, x_2, \ldots, \theta) = \mathcal{N}(\mu_\theta + \alpha_\theta x_1 + \beta_\theta x_2 + \ldots, \sigma_\theta^2)$$

i.e. it is linear Gaussian for each value θ. If $X_1, X_2, \ldots$ are Gaussian, the marginal distribution over Y is a mixture of Gaussians:

$$\begin{aligned}\mathsf{P}(Y) &= \sum_{\theta \in \Theta} \mathsf{P}(\theta)\mathcal{N}(\hat{\mu}_\theta, \hat{\sigma}_\theta^2) \\ \hat{\mu}_\theta &= \mu_\theta + \alpha_\theta \mu_{X_1} + \beta_\theta \mu_{X_2} + \ldots \\ \hat{\sigma}_\theta^2 &= \sigma_\theta^2 + \alpha_\theta^2 \sigma_{X_1}^2 + \beta_\theta^2 \sigma_{X_2}^2 + \ldots\end{aligned}$$

Again, this also holds for complete networks: if all nodes are (conditional) linear Gaussian, the joint distribution is a mixture of multivariate Gaussians. However, this number of components in this mixture is exponential in the number of Θ variables, which can make inference hard. A conditional linear Gaussian can also be written as a deterministic node with extra parents. For this we use a special notation shown in Fig. 2 (right), to which we will return later.

As these three node types can all be written as deterministic nodes, we will henceforth use the convention that *all non-root nodes in our networks are deterministic.*

3.2 LDS models as Bayesian networks

The paper heater model in Fig. 1 translates to the following discrete-time state-space equations in the form of Eq. (1):

$$\begin{aligned}\begin{bmatrix} T_{t+1} \\ T'_{t+1} \end{bmatrix} &= \mathbf{A}_d \begin{bmatrix} T_t \\ T'_t \end{bmatrix} + \mathbf{B}_d \begin{bmatrix} P_t^{\text{in}} \\ 20°\text{C} \end{bmatrix} \\ \mathbf{A}_d &= \begin{bmatrix} 1 - 1/rc & 1/rc \\ 1/rc' & 1 - 1/rc' - 1/r'c' \end{bmatrix} \\ \mathbf{B}_d &= \begin{bmatrix} 1/c & 0 \\ 0 & 1/r'c' \end{bmatrix}\end{aligned} \tag{2}$$

The state of the system consists of the temperatures T_t and T'_t of the two heat masses. In fact, translating the electrical diagram by tools such as 20-sim first leads to a more elaborate form in which auxiliary power variables are present. It is instructive to represent this form as a Bayesian network consisting only of linear deterministic nodes; this is shown at the right side of Fig. 1. As the network is completely deterministic, it might also be read as a system of equations over ordinary variables:

- Each *node* represents the left-hand side of an equation, consisting of one variable.
- The incoming *arcs* represent the right-hand side: a linear combination of the parent variables, with coefficients as specified on the arcs. Note: we follow the convention that empty arcs carry a coefficient of 1.

For example, the figure shows that

$$\begin{aligned}P_t^{\text{trans}} &= \frac{1}{r} T_t + \frac{-1}{r} T'_t = \frac{T_t - T'_t}{r} \\ T_{t+1} &= \frac{1}{c} P_t^{\text{in}} + T_t + \frac{-1}{c} P_t^{\text{trans}} = T_t + \frac{P_t^{\text{in}} - P_t^{\text{trans}}}{c}\end{aligned}$$

The state-space equations (2) are obtained from this system by substituting the P^{trans} and P^{out} variables for their right-hand sides. Interpreted as a Bayesian network, this corresponds to marginalization.

We will now start to add uncertainty to the LDS. First, as is often done (e.g. in the Kalman filter), we augment the state variables with additive zero-mean Gaussian noise:

$$\begin{aligned}\begin{bmatrix} T_{t+1} \\ T'_{t+1} \end{bmatrix} &= \mathbf{A}_d \begin{bmatrix} T_t \\ T'_t \end{bmatrix} + \mathbf{B}_d \begin{bmatrix} P_t^{\text{in}} \\ 20°\text{C} \end{bmatrix} + \begin{bmatrix} W_t \\ W'_t \end{bmatrix} \\ W_t &\sim \mathcal{N}(0, \sigma^2) \\ W'_t &\sim \mathcal{N}(0, \sigma'^2)\end{aligned} \tag{3}$$

The noise is represented by two independent variables W_t and W'_t. A graphical representation of this system is shown in Fig. 3; we have only replaced W_t and W'_t by two anonymous standard Gaussian variables $\mathcal{N}(0, 1)$ whose value is multiplied by σ and σ'. As a result, the T_t variables now have the linear Gaussian form from Fig. 2 (middle).

In fact, this makes the whole system Gaussian: although the P_t^{trans} and P^{out} nodes are not linear Gaussian, we have already seen that they can be marginalized out, making the T_{t+1}, T'_{t+1} nodes directly depend on T_t, T'_t. As for the P_t^{in} node: as it is always used with concrete evidence p_t^{in}, it never represents a probability distribution, and can be reformulated to take

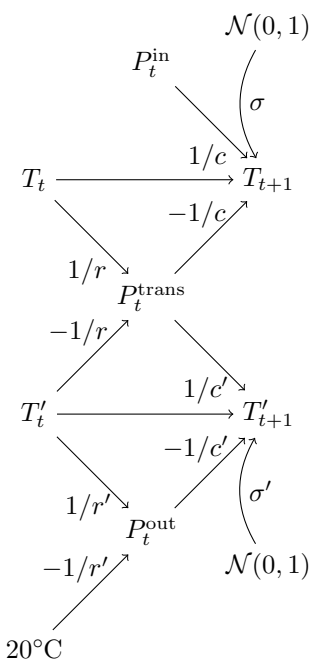

Figure 3: LDS of Eq. (3), containing additive zero-mean Gaussian noise on the state variables.

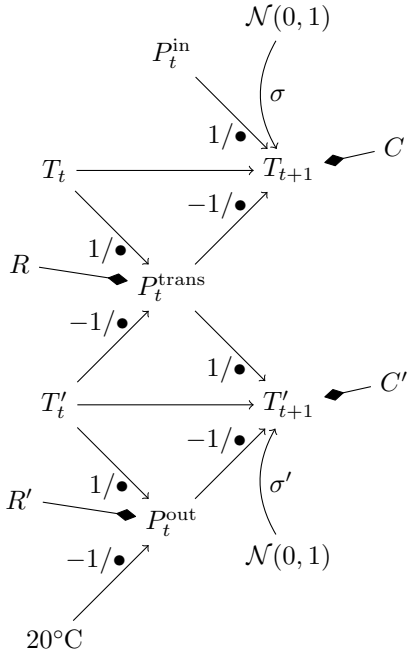

Figure 4: LDS of Eq. (3) augmented with uncertain parameters. These are modelled by *conditional linear deterministic* nodes.

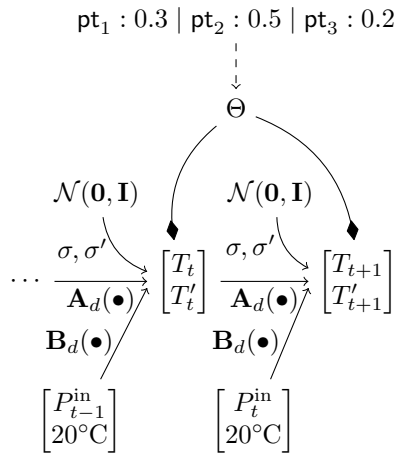

Figure 5: The model from Fig. 4, summarized using vector variables (where Θ represents R, C, R', C') and augmented with a discrete distribution. The relation between time slices $t-1$ and t is also shown. There are several options for the relation between paper type and Θ (the dashed arc here is not a linear influence).

the place of μ in the T_{t+1} distribution. Thus, Fig. 3 is a dynamic Bayesian network with a Gaussian joint distribution. This means that we can use a variant of the standard Kalman filter to do exact inference on this network; we discuss this in detail in Sect. 4.

3.3 LDS models with uncertain parameters

Above, we have shown how to represent an LDS with additive zero-mean Gaussian noise as a Bayesian network; while it may be instructive, thus far it is only a convenient reformulation of known theory. However, to model the relation with the paper type variable, we need uncertainty in the *parameters*, i.e. the coefficients of the linear relation. Our solution is to transform these coefficients into probabilistic variables. We accomplish this by introducing a new node type:

Conditional linear deterministic node: a linear deterministic node extended with a special parent, which is distinguished from the rest because its arc ends in a diamond (and carries no coefficient); we call this parent the *conditioning parent*. The coefficients on the other arcs *can depend on the value of the conditioning parent*. This dependence is shown by putting a bullet in the place where this value is to be filled in.

We have already given an example of such a node: the conditional linear Gaussian node in Fig. 2. Just like the linear Gaussian node is an instance of a linear deterministic node, viz. having specific parents 1 and $\mathcal{N}(0,1)$, a conditional linear Gaussian node is a specific instance of conditional linear deterministic node.

By using conditional linear deterministic nodes, we can extend our already noisy paper heater model with uncertain parameters: we replace parameter r by variable R—which becomes the conditioning parent of the node that depended on r—and do the same for the other parameters. The result is shown in Fig. 4.

We can now proceed to connect a discrete paper type variable to the model, with an example distribution assigning to $\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3$ the probabilities 0.3, 0.5 and 0.2. Like we mentioned, the paper type determines the R' parameter, but for generalization's sake we will assume that it influences all the parameters. The resulting model, in Fig. 5, also shows that the notation for (conditional) linear deterministic nodes extends very naturally to vector-valued variables: coefficients become matrices. These are the matrices from Eq. (2), written as functions $\mathbf{A}_d(\theta)$ and $\mathbf{B}_d(\theta)$ of $\theta = (r, c, r', c')$. Thus, we have made a *graphical summary* of the model which is linked very clearly to the state-space equations. Although this hides some of the model's internal structure, it is useful for keeping an overview.

Regarding the probability distribution of the Θ variable, we give two examples:

Discrete Θ: The paper types $\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3$ deterministically set a value $\theta_1, \theta_2, \theta_3$ (resp.) for Θ. In fact, this turns T_t and T_{t+1} into conditional linear Gaussians (conditioned by the paper type), so the joint distribution is a mixture of 3 Gaussians.

Continuous Θ: The paper types $\mathsf{pt}_1, \mathsf{pt}_2, \mathsf{pt}_3$ determine the parameters $(\mu_{\theta 1}, \sigma_{\theta 1})$, $(\mu_{\theta 2}, \sigma_{\theta 2})$, $(\mu_{\theta 2}, \sigma_{\theta 2})$ for a Gaussian-distributed Θ. This model is no longer linear.

These options have an influence on inference and learning in the model, which we discuss in the next sections.

4 Inference

In this section, we shortly discuss inference in the uncertain parameter model of Fig. 4, for both the discrete and continuous Θ given above. Assume we observe the system's T'_t variable responding to the P^{in}_t input for a while, resulting in data $\mathcal{D} = \{p^{\text{in}}_0, t'_0, \ldots, p^{\text{in}}_{m-1}, t'_{m-1}, t'_m\}$, and the goal is to find out the paper type.

This can be done by a forward pass over the model as known from dynamic Bayesian network literature. We start with the prior distribution

$$\mathsf{P}(t_0, \theta, t'_0) = \mathsf{P}(t_0)\mathsf{P}(\theta)\mathsf{P}(t'_0)$$

and perform a recursive forward pass from $t = 0$ to $t + 1 = m$:

$$\mathsf{P}(t_{t+1}, \theta, p^{\text{in}}_{0..t}, t'_{0..t+1}) = \int \mathsf{P}(t_t, \theta, p^{\text{in}}_{0..t-1}, t'_{0..t})\mathsf{P}(t_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta)\mathsf{P}(t'_{t+1}|t_t, t'_t, \theta)\, dt_t$$

Finally, we marginalize out t_m:

$$\mathsf{P}(\theta, \mathcal{D}) = \int \mathsf{P}(t_m, \theta, \mathcal{D})\, dt_m$$

The details of the inference algorithm depend on the model used. For the discrete Θ, all the distributions above are linear Gaussian, so we can multiply and integrate exactly. To be precise, $\mathsf{P}(t_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta)$ and $\mathsf{P}(t'_{t+1}|t_t, t'_t, \theta)$ are linear Gaussian for each of the 3 individual θ_i values; the algorithm thus independently works on 3 Gaussians.

For the continuous Θ, the conditional distributions of T_{t+1} and T'_{t+1} are not linear Gaussian; we can do *approximate* inference by linearizing these distributions at each timeslice around the means of T_t, Θ (given the data up to t), in analogy to the Extended Kalman Filter (see e.g. [6, 10]).

A second type of inference that we do with the model is *smoothing*. In particular, we want to calculate $\mathsf{P}(\theta, t_t, t_{t+1}|\mathcal{D})$ for each timeslice, in order to do EM learning (see the next section). We have used a *Rauch-Tung-Striebel*-type smoother [9]. This uses a forward pass like discussed above, with the adjustment that it stores the distributions over two time slices. The last of these, i.e. $\mathsf{P}(t_{m-1}, t_m, \theta, \mathcal{D})$, is used as the input for a recursive backward pass defined as follows:

$$\mathsf{P}(t_{t-1}, t_t, \theta, \mathcal{D}) = \int \mathsf{P}(t_t, t_{t+1}, \theta, \mathcal{D})\mathsf{P}(t_{t-1}|t_t, \theta, \mathcal{D})\, dt_{t+1}$$

where the first factor in the integral is the recursive one, and the second is calculated from the distribution over t_{t-1}, t_t stored in the forward pass:

$$\mathsf{P}(t_{t-1}|t_t, \theta, \mathcal{D}) = \mathsf{P}(t_{t-1}|t_t, p^{\text{in}}_{0..t-1}, t'_{0..t}) = \frac{\mathsf{P}(t_{t-1}, t_t, p^{\text{in}}_{0..t-1}, t'_{0..t})}{\int \mathsf{P}(t_{t-1}, t_t, p^{\text{in}}_{0..t-1}, t'_{0..t})\, dt_{t-1}}$$

The advantage of such a smoother over an independent backward pass is that it does not linearize the distribution over T_{t-1}, T in two different ways (the backward pass uses the linearization of the forward pass).

5 Learning

For *learning* the model, we discuss the situation where we *know the paper type* (assume it is pt_1) and observe the system like before, i.e. $\mathcal{D} = \{p^{\text{in}}_0, t'_0, \ldots, p^{\text{in}}_{m-1}, t'_{m-1}, t'_m\}$. The goal is to learn the parameter set ρ that maximizes the likelihood $\mathsf{P}(\mathcal{D}|\mathsf{pt}_1; \rho)$. The situation is a little different depending on the model for Θ.

5.1 EM for continuous Θ

For the continuous Θ, the restriction to pt_1 means that we are learning the parameters for one multivariate Gaussian variable Θ (actually consisting of four independent variables R, C, R', C') and the σ, σ' process noise parameters. Thus, $\rho = (\mu_{\theta 1}, \sigma_{\theta 1}, \sigma, \sigma')$. This can be done by a standard EM algorithm [2] for Bayesian networks: given a set of initial parameters ρ_i, the approximate smoother infers the distributions $\mathsf{P}(t_t, t_{t+1}, \theta|\mathcal{D}, \mathsf{pt}_1; \rho_i)$. From these, the expected sufficient statistics are gathered for maximizing

$$\mathsf{P}(\mathcal{D}, T_{0..m}, \Theta|\mathsf{pt}_1; \rho_{i+1})$$

expected under the old parameters ρ_i; this is repeated until convergence.

5.2 EM for discrete Θ

For the discrete parameter space model, we are looking for the parameter set $(\theta, \sigma, \sigma')$ that maximizes the likelihood

$$\mathsf{P}(\mathcal{D}|\mathsf{pt}_1, \Theta = \theta; \sigma, \sigma')$$

Note that the role of Θ is different here; we are not learning the optimal parameters for a distribution over Θ, but the optimal single value. This requires some adjustments to the smoother: it should store distributions over (T_t, T_{t+1}) instead of over (T_t, T_{t+1}, Θ), and should not use a prior distribution over Θ either. Because all the probability distributions are linear Gaussian again, smoothing is exact now.

However, maximizing the expected likelihood is not so trivial now: we are looking for the optimal linear Gaussian distribution $\mathsf{P}(t_{t+1}, t'_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta)$ *constrained to a certain form* prescribed by $\mathbf{A}$ and $\mathbf{B}$. Specifically, the log likelihood for an individual time slice is:

$$\log \mathsf{P}(t_{t+1}, t'_{t+1}|t_t, t'_t, p^{\text{in}}_t, \theta) = -\frac{1}{2}\delta^T \Sigma^{-1} \delta - \frac{1}{2}\log|2\pi\Sigma|$$

where $\Sigma = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma'^2 \end{bmatrix}$ and δ is an abbreviation for

$$\delta = \begin{bmatrix} t_{t+1} \\ t'_{t+1} \end{bmatrix} - \mathbf{A}_d(\theta) \begin{bmatrix} t_t \\ t'_t \end{bmatrix} - \mathbf{B}_d(\theta) \begin{bmatrix} p^{\text{in}}_t \\ 20°\text{C} \end{bmatrix}$$

Separating variables and parameters, we can write:

$$\begin{aligned} \delta &= \mathbf{D}(\theta)\mathbf{x}_t \\ \mathbf{D}(\theta) &= \begin{bmatrix} \mathbf{I} & -\mathbf{A}_d(\theta) & -\mathbf{B}_d(\theta) \end{bmatrix} \\ \mathbf{x}_t &= \begin{bmatrix} t_{t+1} & t'_{t+1} & t_t & t'_t & p^{\text{in}}_t & 20°\text{C} \end{bmatrix}^T \end{aligned}$$

The log likelihood for all the time slices (ignoring the term $-\frac{1}{2}\log|2\pi\Sigma|$ for now) is then:

$$\log \mathsf{P}(\mathcal{D}, t_{0..m}|\theta) = -\frac{1}{2}\sum_{t=0..m} (\mathbf{D}(\theta)\mathbf{x}_t)^T \Sigma^{-1} \mathbf{D}(\theta)\mathbf{x}_t$$

$$= -\frac{1}{2}\sum_{t=0..m} \frac{(\mathbf{D}_1(\theta)\mathbf{x}_t)^T \mathbf{D}_1(\theta)\mathbf{x}_t}{\sigma^2} + \frac{(\mathbf{D}_2(\theta)\mathbf{x}_t)^T \mathbf{D}_2(\theta)\mathbf{x}_t}{\sigma'^2}$$

where $\mathbf{D}_i$ denotes the ith row of $\mathbf{D}$. The goal is to maximize the expected value of this expression. At first sight, it seems that the two terms are dependent through θ, but on closer inspection

$$\mathbf{D}(\theta) = \begin{bmatrix} 1 & 0 & -1+1/rc & -1/rc & -1/c & 0 \\ 0 & 1 & 1/rc' & -1+1/rc'+1/r'c' & 0 & -1/r'c' \end{bmatrix}$$

we see that the values in the first row do not constrain those in the second, or vice versa. We can therefore minimize the expected value of the two terms independently. We can also see that there *are* linear constraints for the values *within* a row, e.g. $\mathbf{D}_{1,3}(\theta)+\mathbf{D}_{1,4}(\theta) = -1$. We record these constraints in a matrix $\mathbf{C}$ and vector $\mathbf{c}$ such that $\mathbf{C}\mathbf{D}_1^T(\theta) = \mathbf{c}$. Substituting $\mathbf{d} = \mathbf{D}_1^T(\theta)$, for the first term we are looking for the $\mathbf{d}$ that minimizes

$$\sum_{t=0..m} E(\mathbf{x}_t^T \mathbf{d}\mathbf{d}^T \mathbf{x}_t) = \mathbf{d}^T \left[\sum_{t=0..m} E(\mathbf{x}_t \mathbf{x}_t^T) \right] \mathbf{d}$$

under the constraint $\mathbf{C}\mathbf{d} = \mathbf{c}$. This is a linearly constrained quadratic optimization problem that can be solved by the method of Lagrange multipliers. The second term can be minimized in the same way.

In conclusion, we have derived the M-phase for the discrete Θ model; in the E-phase, we therefore have to collect the expected sufficient statistics $E(\mathbf{x}_t \mathbf{x}_t^T)$.

5.3 Comparison

It is interesting to compare learning for continuous and discrete Θ. In order to do this, we have simulated the system in Fig. 1 for 150 time slices, with a sine wave as P^{in} input and random Gaussian disturbance. We provide the EM algorithms discussed above with the P^{in} and the generated T'_t data. Typical results of a 60-iterations run are shown in Fig. 6.

The most interesting fact to observe is that the two approaches converge to different values (but the same log likelihood). This probably means that the system is not *identifiable*: several choices for Θ lead to the same likelihood of the observed behaviour T'_t. To test this hypothesis, we also generated synthetic T_t, T'_t data (without disturbance) for systems with the learned parameters. The results are plotted in Fig. 7. These results indeed show that both methods arrive at the same approximation (green, red) of the original (blue) T'_t data; however, the different values for the parameters lead to a different behavior of T_t.

A second observation from Fig. 6 is that learning continuous Θ converges faster than learning discrete Θ. The explanation for this is as follows: the EM algorithm for the continuous parameter space uses inference to compute a posterior distribution over the variable Θ. In this algorithm, the posterior distribution is updated for each time slice. However, we can also regard the algorithm as doing full Bayesian learning where Θ is viewed as a parameter; the algorithm is then performing *incremental EM* [8], which is known to converge faster.

6 Conclusion

The central scientific hypothesis which initiated the research described in this paper was that knowledge engineering methods for industrial applications of probabilistic graphical models should be based as much as

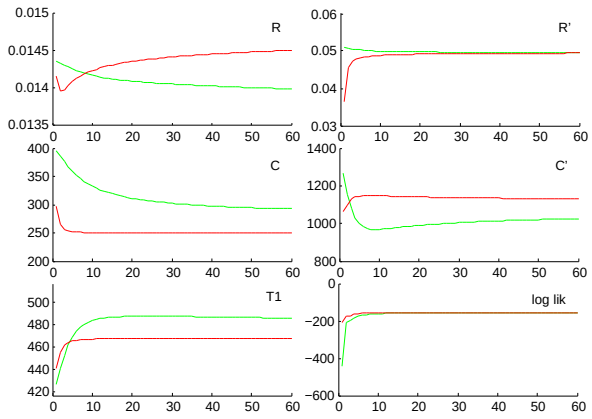

Figure 6: EM learning of the Θ parameters: comparison of discrete parameter space (parameters are single values; shown in green) and continuous parameter space (parameters are Gaussians; μ values shown in red). The horizontal axis represents 60 EM iterations. Also shown are the learned distribution over T_1 (Gaussian, μ value) and the log likelihood.

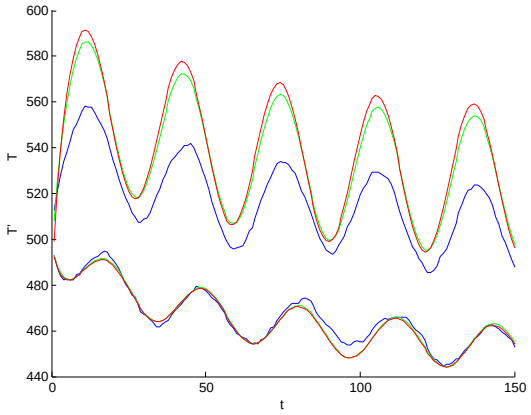

Figure 7: Blue: synthetic data used for learning (T_t and T'_t are shown, but only the latter is given to the learning algorithms). As input P_t^{in}, we used a sine wave. We disturbed the system with additive zero-mean Gaussian noise. **Green, red**: response of an undisturbed deterministic system using the learned parameters (with discrete and continuous parameter space, resp.) to the same P_t^{in} sine wave.

possible on existing methods from engineering. We have developed a systematic framework, where we start with linear system theory and associated diagrams and notations as the first step in the knowledge engineering process. The advantage of this approach is that engineers have already dealt with the unavoidable complexity issues in representing realistic models. Subsequently, it was shown how linear dynamic system models can be augmented with probabilistic variables for uncertain parameters, transforming them into dynamic Bayesian networks with *conditionally linear* nodes. We introduced a concise notation that combines LDS and Bayesian network concepts in a natural way and demonstrated methods for inference and learning from data in these models. The practical usefulness of the framework was illustrated by a case study from the domain of large production printers.

Acknowledgements

This work has been carried out as part of the OCTOPUS project under the responsibility of the Embedded Systems Institute. This project is partially supported by the Netherlands Ministry of Economic Affairs under the Embedded Systems Institute program.

References

[1] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[3] A.J. Hommersom and P.J.F. Lucas. Using Bayesian networks in an industrial setting: Making printing systems adaptive. In *19th European Conference on Artificial Intelligence*, pages 401–406, 2010.

[4] R.E. Kalman et al. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[5] D.C. Karnopp, D.L. Margolis, and R.C. Rosenberg. *System dynamics: modeling and simulation of mechatronic systems*. John Wiley & Sons, 2006.

[6] D. Koller and N. Friedman. *Probabilistic graphical models: Principles and techniques*. The MIT Press, 2009.

[7] K.B. Korb and A.E. Nicholson. *Bayesian Artificial Intelligence*. Chapman & Hall/CRC, 2004.

[8] Radford Neal and Geoffrey E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M.I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[9] H.E. Rauch, F. Tung, and CT Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450, 1965.

[10] H.W. Sorenson. *Kalman filtering: theory and application*. IEEE, 1985.

[11] `http://www.20sim.com/`.