TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta strojní

Identification of dynamic systems using Bayesian networks

Disertační práce

Studijní program:	P2301 Strojní inženýrství
Studijní obor:	Výrobní systémy a procesy
Autor práce:	**Ing. Maryna Garan**
Školitel práce:	doc. Ing. Petr Lepšík, Ph.D. Katedra částí a mechanismů strojů

Liberec 2019

Prohlášení

Byla jsem seznámena s tím, že na mou disertační práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé disertační práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li disertační práci nebo poskytnu-li licenci k jejímu využití, jsem si vědoma povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Disertační práci jsem vypracovala samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé disertační práce a konzultantem.

Současně čestně prohlašuji, že texty tištěné verze práce a elektronické verze práce vložené do IS/STAG se shodují.

31. října 2019 Ing. Maryna Garan

Abstract

The aim of this thesis is to provide the bridging between Bayesian networks and system identification. Firstly, the literature review and necessary theoretical prerequisites are provided. Secondly, Bayesian network based models of dynamic systems are introduced. Next, the methodology of Bayesian network based system identification is proposed and explored on simulated datasets. In addition, a new approach to the order selection for a resulting model is proposed and verified. Finally, the proposed Bayesian network based system identification approach is verified on real dynamic systems.

Overally, the thesis proposes a new approach to system identification of dynamic systems influenced by noisy signals. In addition, Bayesian network based models proposed in this thesis can be used for state estimation, monitoring and control of dynamic systems.

Keywords: Bayesian networks, system identification, order selection, dynamic system

Abstrakt

Cílem této práce je vytvoření spojení mezi Bayesovskými sítěmi a parametrickou identifikací dynamických systémů. Nejprvé byl zpracován průzkum dostupné literatury a byly zformulovány důležité teoretické základy. Poté jsou uvedeny modely dynamických systémů na bázi Bayesovských sítí. Těžištěm práce je návrh a ověření metodologie identifikace dynamických systémů pomocí Bayesovských sítí. Součástí metodologie je nový přístup k volbě řádu výsledného modelu. Na závěr, byla ověřena navržená metoda identifikace dynamických systémů pomocí Bayesovských sítí na fyzikálních modelech dynamických systémů.

Obecně je možno konstatovat, že je disertační práce zaměřena na návrh nového přístupu k identifikaci dynamických systémů ovlivněných šumem. Uvedené modely dynamických systémů na bázi Bayesovských sítí mohou být také využité k estimaci stavu, sledování a řízení dynamických systémů.

Klíčová slova: Bayesovské sítě, identifikace, volba řádu, dynamický systém

Acknowledgements

First and foremost, I would like to thank to my former supervisor, Miroslav Olehla, who unpredictably deceased a year ago. Also, I would like to thank to my current supervisor, Petr Lepšík, for his support and encouragement that helped me to finish this thesis.

I was very lucky to meet many talented, interesting and inspiring people during my doctoral study. It would not be possible to list all of them, but there are several, which I have to mention personally. I would like to thank to Michal Moučka for providing me with the opportunity to experience doctoral study adventure and for being a good friend along the way. I am very thankful to Sylvain Verron for being my supervisor during two unforgettable internships in France. He introduced me to Bayesian network framework and accompanied me while I was getting acquainted with it. I would like to thank to Osvald Modrlák for his contagious love for control theory and for people. It would be very difficult for me to finish this thesis without his help and encouragement, in particular, during the last and the toughest year of my study. Also, I would like to thank to Petr Volf for all those pleasant hours spent on discussions around probability theory and statistics.

I would also like to thank to all my colleagues and university members, who made my doctoral study unique and unforgettable.

From the bottom of my heart, I would like to thank to my parents and my brother for their love, constant support and for giving me the opportunity to go abroad and to explore my potential.

And finally, I would like to thank to the best teammate I could ever dreamed of, my husband. Thank you for being by my side during failures with a cup of hot tee and during victories with a glass of red wine.

Contents

List of abbreviations and symbols

List of abbreviations

(n+1)TDBN	(n+1) time slices based dynamic Bayesian network
2TDBN	Two time slices based dynamic Bayesian network
AIC	Akaike information criterion
ANN	Artificial neural network
BIC	Bayesian information criterion
BN	Bayesian network
BNSI	Bayesian network based system identification
BNT	Bayes Net Toolbox
CPD	Conditional probability distribution
CPT	Conditional probability table
DAG	Directed acyclic graph
DBN	Dynamic Bayesian network
DCG	Directed cyclic graph
DED	Discrete event dynamic system
DNA	Deoxyribonucleic acid
EKF	Extended Kalman filter
EM	Expectation maximization
HIV	Human immunodeficiency virus
HMM	Hidden Markov model
KF	Kalman filter
LDS	linear dynamic system

LL	likelihood
MAP	Maximum a posteriori
MIMO	Multi input multi output
MISO	Multi input single output
MLE	Maximum likelihood estimate
MSE	Mean squared error
MUNIN	Muscle and nerve inference network
N4SID	Numerical algorithms for subspace state space system identification
PEM	Prediction error method
QMR-DT	Quick medical reference, decision theoretic
SHM	Structural health monitoring
SI	System identification
SIMO	Single input multi output
SISO	Single input single output
SPC	Statistical process control
UKF	Unscented Kalman filter

List of symbols

$\lambda_0, \lambda_1, \cdots, \lambda_n$	Parameters of linear regression
$\mathbb{A}, \mathbb{B}, \mathbb{C}, \mathbb{D}$	State matrices
$\mathbb{I}$	Unitary matrix
$\mathbb{K}$	Kalman gain
$\mathbb{L}$	Luenberger observer gain
$\mathbb{P}$	Auxiliary matrix for recalculation of initial conditions
$\mathbb{P}$	Error covariance
$\mathbb{S}$	Auxiliary matrix for recalculation of initial conditions
$\mathbb{U}(t_0)$	Matrix of initial conditions for the input signal
$\mathbb{X}$	State vector
$\mathbb{Y}(t_0)$	Matrix of initial conditions for the output signal

$\mathcal{L}\{\cdots\}$	Laplace transform
$\mathcal{Z}\{\cdots\}$	Z-transform
μ	Mean of a random variable
$\omega_1, \omega_2, \cdots, \omega_n$	Weights in mixtures of normal disributions
Σ	Covariance matrix of a multivariate random variable
σ^2	Variance of a random variable
σ^2_ϵ	Variance of white noise in simulated stochastic systems
σ^2_{exp}	Values of fixed variance used in learning scenarios 2,3,4 and 6
$a_0, a_1, \cdots, a_n$	Coefficients associated with derivations of the output signal
$b_0, b_1, \cdots, b_m$	Coefficients associated with derivations of the input signal
Dim	The dimension of a directed acyclic graph
dt	Sampling rate, discretization step
$G(s)$	Transfer function (for continuous-time models)
$G(z)$	Transfer function (for discrete-time systems)
J	Cost function
M	The number of training examples
m	Order of the highest derivation of the input signal
n	Order of a dynamic system
s	Complex variable used in Laplace transform
T	The length of a signal
u	Input signal
$U(s)$	Laplace transform of the input signal
$U(z)$	Z-transform of the input signal
$x_1, x_2 \cdots, x_n$	State variables
y	Output signal
$Y(s)$	Laplace transform of the output signal
$Y(z)$	Z-transform of the output signal

List of Figures

List of Tables

Introduction

Since ancient times, people try to explain phenomena that surround us in the world and to find regularities in their appearance. Mathematics provided us with comprehensive toolbox to fulfill these tasks. In the middle of the previous century one of the main problems that slowed down scientific research was the necessity in difficult mathematical calculations. After invention and spread of computers this problem was eliminated and it gave a great impetus to many scientific fields. Moreover, it gave the second life to many tools, methods and algorithms that were hard or even impossible to implement because of their complicatedness. Bayesian networks are the representatives of such tool.

Bayesian networks allow reasoning about random variables under the influence of uncertainties. They can provide relatively compact representation of the joint probability distribution over enormous number of random variables (discrete, continuous or the mixture of both) and are particularly useful in practice due to their ability to incorporate expert knowledge in a model and to cope efficiently with partially observed data [1].

System identification is a scientific field that incorporates methods for discovering appropriate mathematical description of dynamic processes that is crucial for the simulation of their behavior and for the design of efficient controllers. System identification as a scientific field separated from control theory in 1960th. Since then, there have been proposed a lot of identification methods that use distinct models and approaches to describe the behavior of dynamic systems. It is important to point out, that there are no such terms as a “good” or “bad” method, each of them has certain advantages, disadvantages and restrictions. Hence, it is crucial to consider the character of a dynamic system, the type of a task the identification procedure has to be provided for and the precision required to successfully carry out this task during choosing of the most appropriate method [2].

In system identification field, dynamic systems can be treated as deterministic or stochastic. In the former case, the output is assumed to be unequivocally determined by the parameters of a system and by the sequence of input signals. In the latter case, a dynamic system is assumed to be influenced by random noise. Bayesian networks based algorithms for system identification will broaden the set of stochastic methods with a new member.

One can ask, why do we need to use stochastic models, which are more complicated to work with? There are three basic reasons why deterministic dynamic models often do not provide sufficient means of performance. Firstly, no mathematical model is perfect, each one has many sources of uncertainties. Secondly, the behavior of dynamic systems is influenced not only by control inputs, but also by disturbances which cannot be modeled deterministically. And last but not the least, the information about signals comes from sensors that do not provide complete and perfect data [3]. In addition, Bayesian networks can cope with partially missing data and with partially known system structure and parameters.

Bayesian networks have been successfully used in control systems engineering for monitoring [4, 5], system control [6, 7, 8], fault detection and diagnosis [9, 10]. Since solving of these tasks in many cases requires a model of investigated dynamic system, the system identification procedures have to be provided as a preliminary step. The ability of Bayesian networks to serve as a system identification tool was mentioned in several publications [11, 12, 13], but this task has not yet been addressed in available literature and research articles. Therefore, solving of this task was chosen as the main objective of this doctoral thesis.

The potential of Bayesian networks as a tool that can be used in dynamic system modelling for the description and inference of signal flows was emphasized by Lennart Ljung, a leading researcher in control theory, in his recent article “Perspectives on system identification” [14]. He also emphasizes the increasing need of knowledge exchange between different research areas, that will particularly help to enrich the set of available system identification tools.

This thesis aims to fill the gap between dynamic systems seen with the perception of control system engineers and Bayesian network framework. In particular, the main goal is to discover the performance of Bayesian networks in solving the task of system identification. As a result, Bayesian networks may be used as a unified tool for control-related tasks.

Obtained results may also be used for fault diagnosis purposes. As a rule, Bayesian networks are considered the representatives of data-driven approaches to fault diagnosis [15]. In contrast, the approach to system modelling considered in this thesis is model-based. Therefore, the proposed models can be used for implementation of model-based fault diagnosis approaches, e.g. BN-based parameter estimation. Also, it can be used in combination with other methods to implement so-called hybrid approaches to fault diagnosis, refer to [15] for more details on the topic.

The structure of the thesis is as following. Literature review is presented in the chapter 1, where the review of system identification methods (section 1.1), short history of Bayesian networks, the overview of their recent implementations (section 1.2) and the analysis of the state of the art in the interconnection between Bayesian networks and control systems engineering (section 1.3) are presented. The chapters 2 and 3 provide theoretical preliminaries essential for understanding the further chapters. The thesis is meant to be understandable for both control engineers and statisticians, hence the theoretical part contains short introduction to dynamic system modelling from the control theory perspective (chapter 2) and to Bayesian network framework (chapter 3). Models of dynamic systems based on Bayesian networks that can be used for modelling and system identification are presented in the chapter 4. The chapter 5 provides the results of practical experiments. The methodology of Bayesian network based system identification (BNSI) is proposed in the section 5.1. Detailed description of experiments provided for its verification and the algorithm of their evaluation are addressed in the section 5.2. Experiments aimed to find the optimal setting of tuning parameters are described in the sections 5.3, 5.4 and 5.5. The verification of the proposed system identification algorithm is provided on the simulated responses of stochastic dynamic systems in the section 5.6, order selection approach is presented and verified in the section 5.7. The results of the implementation of BNSI for identification of real dynamic systems are presented in the chapter 6.

1 Literature Review

Since the considered topic of research lies in the overlapping between two different scientific fields, it was necessary to provide the literature review in both fields and then present the state of the art in their interconnection.

The section 1.1 is dedicated to the field of system identification. In this section, the most prominent methods used for the identification of dynamic systems are reviewed. The section 1.2 is dedicated to Bayesian networks. The section starts with the historical review that explains why Bayesian networks have become popular relatively recently. Also, the overview of recent applications in different scientific areas is provided.

Since interconnection between Bayesian networks and system identification has not been addressed in available literature, the state of the art in the section 1.3 is presented from the broad perspective of control systems engineering. This viewpoint was chosen since system identification is the subfield of control engineering that provides other subfields (e.g. control, monitoring) with models of dynamic systems given the measurements provided on those systems.

1.1 Review of Identification Methods

A dynamic system is an object that produces observable signals that depend on the interactions between different internal variables, previous values of these variables and external stimuli. Analyzing and usage of dynamic systems require knowing of their behavior, which is commonly described by a mathematical model [16]. There are two basic approaches for obtaining the model of a dynamic system: mathematical modeling (analytical approach) and system identification (experimental approach) [17]. The former one is based on splitting of the system into subsystems, whose behavior and properties are known, and on binding these subsystems mathematically into the model that describes behavior of the entire system [16]. Mathematical modelling often does not require any measurements on a real system. On the other hand, this approach may be too complicated for complex systems and it requires extensive prior knowledge about a technological process. Mathematical modelling is not addressed in this thesis, refer to [18] for more details on this topic.

System identification (SI) is the process of building the mathematical model of a dynamic system based on data observed from a system [16]. These models can be used for simulation, control systems design, monitoring, fault detection, quality control, etc. They are highly useful for systems that are difficult to experiment with (due to the expensiveness of experiments or the danger that they cause) [19]. While mathematical modeling provides a description that explains underlying essential mechanisms using physical laws (which may be interesting for physicians), system identification provides a model that is more useful for practical applications (which may be interesting for engineers). However, considering the character of an application, it is also often required to trade-off model complexity versus accuracy [19].

The challenge of obtaining mathematical model of a technological process from measurement has interested scientific community for a long time. The term "identification" for procedures that face this challenge was firstly proposed by Lofti Zadeh in 1956 [20]. System identification as an independent field separated from control theory in sixtieth, its development has been constantly supported by the IFAC symposia on Identification that has been organized since 1967 once in three years [2]. The overview of classical system identification methods can be found in [16, 17]. In addition to these iconic books it is also worth to mention further informative publications, like [21, 22] or more recent ones [2, 23, 24, 25].

The overview of identification methods in modern control theory requires introduction of several important classification criteria. Firstly, we have to choose a type of mathematical model that will be used in identification procedure. This model reflects a functional dependence between input and output variables, sometimes internal variables of a system are also taken into account. Also, this model can be expressed either in a form of mathematical equations (parametric model) or in the form of graphs, respectively tables that can be used to build them (non-parametric model). In the former case, we assume that the behavior of a system can be approximated by the model of a certain structure with finite number of explicit parameters. In this case, the identification task reduces to the estimation of the unknown parameters of a known model. In the latter case, both parameters and a structure are unknown, parameters of a system are implicitly included in the model. These models can also be viewed as models with infinite number of parameters [2].

Depending on the amount of inputs and outputs of a dynamic system, one can distinguish single-input single-output (SISO) systems and multi-input multi-output (MIMO) systems. Depending on the type of these signals we can analyze continuous-time or discrete-time systems. In simple settings, dynamic systems are assumed to be linear (meaning that the steady-state of output is a linear function of the corresponding excitation) and time-invariant (meaning that their parameters are constant). More sophisticated systems can have non-linear behavior, and/or their parameters change over time (time-variant systems), and therefore they have to be treated accordingly.

And last but not the least, the type of interconnection between the process of our interest (dynamic system) and evaluation unit (typically, computer) influences the range of identification algorithms that can be used for a considered process. If a dynamic system is not coupled with a computer, identification procedure requires gathering of measured data, storing and subsequent evaluation. This type of evaluation is called offline identification. In contrast, when coupling between a dynamic system and a computer allows real-time evaluation (in parallel with measurements), we speak about online identification.

Overview of the most prominent identification methods is given below, see figure 1.1. Since there is no standard classification of system identification methods, the overview is provided according to the best beliefs of the author based on available literature. Some methods are described in more details, since they will be referred to later in the thesis, others are given only in introductory manner for completeness of the overview.

Non-parametric methods

Fourier analysis is a method that can be used for linear time-invariant SISO or MIMO dynamic systems in both offline and online settings [2]. This method is used for obtaining frequency response from the step or impulse response of a dynamic system. Spectral analysis of non-periodic signals using Fourier transformation serves for this purpose [16, 17, 2, 26].

Frequency response measurement is a method that can be used for linear time-invariant SISO or MIMO dynamic systems in offline setting [2]. This method is based on the measurement of the responses of a dynamic system on periodic signals with different frequencies. Often measurement procedure for this identification method is quite time-consuming. For systems with low disturbances this method works particularly well, in the case of larger disturbances frequency response measurement with correlation functions can be used for performance improvement [16, 17, 2, 26].

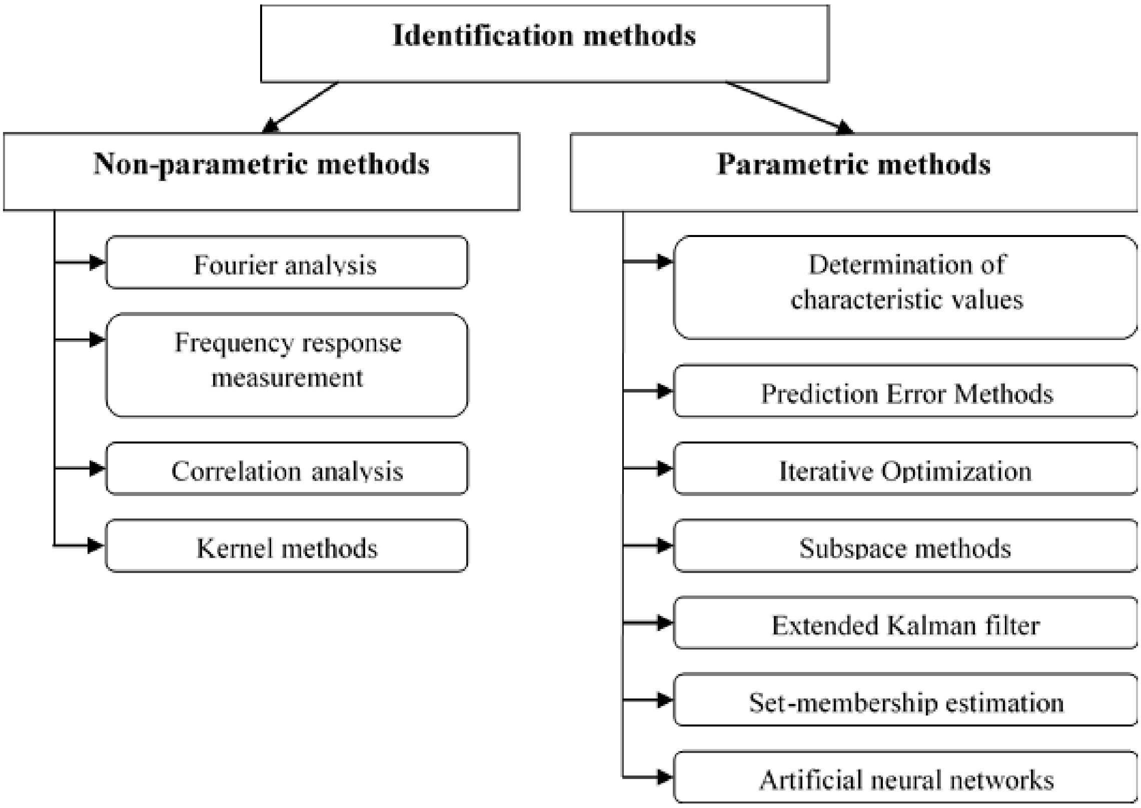


Figure 1.1: Overview of identification methods

Correlation analysis is a method that can be used for linear SISO or MIMO time-variant or time-invariant dynamic systems in both offline and online settings [2]. This method is provided in time domain, both periodic and stochastic signals can be used as test signals, the resulting models are correlation functions. In special case, if pseudo-random binary signals are used as test signals, correlation analysis allows to directly identify impulse responses [16, 17, 2, 26].

Kernel methods were adopted from machine learning due to their ability to trade-off model complexity versus accuracy. Considering this problem, often also called bias/variance trade-off, is crucial for effective implementation of machine learning algorithms. If algorithm has high variance, it overfits training data and the resulting estimator performs well on the training data, but fails to generalize over unseen data (for example, over a cross-validation set). This problem is often caused by an over-complicated model. On the opposite side, the high bias of an algorithm mostly is due to an over-simplified model and consequently, an algorithm fails to fit well even a training set [27]. In system identification, bias/variance problem appears when we choose model complexity (e.g. structure, order). Kernel methods offer distinct approach to the task of system identification than in “traditional” techniques, e.g. described in [16, 17]. Recent reviews of the usage of Kernel methods in system identification can be found for example in [28, 29].

Kernel methods bypass difficulties caused by the selection of model structure and its order by the introducing of a non-parametric form of a utilized model structure. These models include kernel functions, which determine the hypothesis space for an estimation problem. The type of kernel function used in identification procedure defines the amount of prior knowledge that can be incorporated into a model. Newly introduced kernel functions can incorporate, e.g., smoothness, damping, resonance behavior, stability, etc. [30]

The model of an estimator can be defined in one of two formulations: deterministic or probabilistic. In the former case we consider regularization perspective and in the latter case Bayesian perspective [28]. Therefore, system identification based on kernel methods is often referred to as regularized or Bayesian system identification [28, 29, 30].

Parametric methods

Determination of characteristic values is the simplest identification method that can be used for linear SISO time-invariant dynamic systems in offline setting. Characteristic values (e.g. transport delay, time constant) can be determined from the step or impulse response of a system with the aid of diagrams and tables. This method can be used only for simple processes with small disturbances and it is not precise. However, it can be used as a starting point for more sophisticated methods, for example for the rough estimate of time constants [2].

Prediction Error Methods (PEMs) is the wide set of parametric methods that can be used for broad range of dynamic systems (linear or non-linear, SISO or MIMO, time-invariant or time-variant) in both offline and online settings [2]. These methods use differential (for continuous-time systems) and difference (for discrete-time systems) equations that can be extended by a transport delay.

PEMs were the first class of parametric methods used in system identification. These methods are based on the minimization of error signals by the means of statistical methods. It was proven, that under the assumption that noise is normally distributed with zero mean, the Maximum Likelihood estimates (MLEs) of parameters can be obtained from noisy measurement by minimization of a cost function in the form of the sum of squared errors [31]. This leads to a well-known least square method for parameter estimation. This type of a cost function is convex and as a result, has just one (global) minimum. On the other hand, it overemphasizes outliers (since errors are squared).

Following development of the system identification field contributed to appearance of the further modifications and alternative solutions of the least square parameter estimation [2]:

- recursive least squares
- least squares with correlation function
- recursive least squares with correlation function
- weighted least squares
- generalized least squares
- extended least squares
- method of bias correction
- total least squares
- instrumental variables
- method of stochastic approximation
- normalized least squares
- least squares for frequency response approximation

The challenging issue in application of PEMs is that of the selection of model complexity (e.g. choosing an appropriate order for differential/difference equation of a dynamic system). This issue can be addressed by cross-validation methods or by penalized criteria. In the former case, the performance of different models is compared on the cross-validation dataset, a set of measurements from a system that was not used in identification procedure. In the latter case, the optimal model is found by the optimization of a chosen penalized goodness-of-fit criterion [32]: Akaike information criterion (AIC) [33], Bayesian information criterion (BIC) [34], etc. Resulting estimators are referred in literature as Post-Model-Selection Estimators [35].

Iterative optimization methods can be used for time-invariant (SISO or MIMO) dynamic systems in offline setting. These methods can use various cost functions (including the functions that are not linear in parameters), consequently they can cope with non-linear systems. Moreover, important constraints (e.g. stability of a dynamic system) can be included in a cost function. In addition, iterative methods can be used for optimization problems that do not have solutions in closed form [2].

Although iterative optimization procedures propose a wide range of possible implementations, they also have certain challenges to face with. The main disadvantage is that convergence of these methods cannot be guaranteed. It is caused by the fact that cost functions are not guaranteed to be convex, consequently they are susceptible to have local minima. In addition, in many cases iterative methods are computationally demanding [2].

Subspace methods can be used for linear time-invariant dynamic systems in offline setting [2]. These methods are based on the state space representation of dynamic systems, which offers intuitive extension from SISO systems to MIMO systems. These methods are based on Singular Value Decomposition and Least-Square techniques and provide semi-automatic model order determination [19, 2]. The most prominent subspace methods for identification are [19]:

- **N**umerical algorithms for **S**ubspace **S**tate **S**pace **S**ystem **ID**entification (N4SID)
- Multivariable output-error state space
- Canonical variate analysis

The implementation of subspace methods can be challenging, since they involve large computational efforts. In addition, state variables are in most cases immeasurable and often non-interpretable. Moreover, these methods, by themselves, are not suitable for identification in closed loop due to correlations between input and output variables caused by feedback [2]. However, the latter disadvantage can be addressed by extended subspace methods, for example by orthogonal decomposition [36], innovation estimation method, whitening filter approach [37], canonical correlation analysis [38] or others.

Extended Kalman filter (EKF) is a parametric method that can be used for the wide range of dynamic systems in offline and online identification [25]. Originally the Kalman filter (KF) [39] was designed as a state space based model that can be used, depending on a setting, for filtering, smoothing or prediction. The one-step ahead prediction problem is a typical setting for state variable estimation [2].

The original Kalman filter was designed for time-invariant discrete linear systems under the assumption that state variables and input variable are normally distributed [39]. Later the formulation of Kalman filter was extended also for time-variant systems. Its implementation in continuous-time setting can be provided in two ways: by discretizing and consequent application of classical discrete Kalman filter or by a special continuous-time extension of Kalman filter, called Kalman-Bucy filter [40].

Kalman filter can be used for system identification either as a state estimator in combination with subspace methods or directly by application of so-called Extended Kalman filter. The latter is a re-formulation of ordinary Kalman filter, in which both states and parameters of a system are calculated. It is important to mention, that parameters in this case are treated (similarly to state variables) as being influenced by stochastic disturbances [2].

Application of the KF for non-linear systems can be challenging. One of the possible ways to solve this task is to use EKF for linearization and apply traditional linear Kalman filter equations afterwards. The alternative way is to use so-called Unscented Kalman filter (UKF), a formalism that was designed specifically for non-linear dynamic systems. The UKF has superior implementation properties, since it does not require preliminary linearization and there is no need to calculate Jacobians. In addition, it has higher performance and weaker initial assumptions (noise is not assumed to be normally distributed) [41]. Moreover, this method can be easily applied for both state and parameter estimation [42].

Set-membership estimation methods are considered as methods of control-oriented system identification, meaning that they aim to meet requirements of robust control design [43]. As opposite to the classical (statistical) estimation of parameters, for which noise is represented as a stochastic signal, in set-membership estimation it is represented as an unknown but bounded deterministic signal [44]. While the statistical estimation deals with an average case, deterministic estimation considers the worst case, meaning that the estimate shows the best performance in the worst-case setting [45]. Therefore, this approach is also called the worst-case/deterministic approach to system identification [43]. Review of these methods and implementation notes can be found, for example, in [44, 45, 43].

Artificial Neural Network (ANN) is a universal approximator for static and dynamic non-linearities, therefore it is widely used for identification of non-linear dynamic systems [46]. These models require little to no prior knowledge about the structure of a model and can be intuitively extended to MIMO case. These models can be used in both time-invariant [2] and time-variant [47] cases, in both offline [2] and online [48] settings.

ANN consists of neurons that are connected by links (feedforward, feedback, recurrent or lateral). Each neuron is represented by an input operator (e.g. scalar product, Euclidean distance) and activation function (mostly non-linear, e.g. sigmoid, tangent hyperbolic, Gauss) connected in series. Neurons are arranged into layers, a network consists of one input layer, one or more hidden layers and one output layer. The wide range of choices leads to the plenty of possible final structures and, consequently, the plenty of non-linearities that can be caught by a network [2].

Parameter searching procedure in neural networks consists of two steps: training and generalization. In the first step, parameters of neural network (also called weights) are estimated from measured data (this process is often called “learning”). In the second step, network with obtained parameters is used to simulate new data. The goal is to obtain a network with the smallest possible error for both training and generalization [2].

Considerable disadvantage of ANNs in system identification is that the parameters of a network often cannot be interpreted in physical sense [2]. However, overcoming of this restriction is a task addressed in research articles. Refer, for example, to [49] for an algorithm that transforms neural network with known parameters into the transfer function of a dynamic system.

The identification task requires special structures of ANN to capture the dynamic character of underlying process. The aim is to extend standard static ANNs, for example Multi Layer Perceptron, Radial Basis Function or NF Neuro-Fuzzy, to dynamic case. Dynamic neural networks are obtained either by adding external dynamic elements (neural networks with external dynamics) or by the incorporating of dynamic elements within the model structure (neural networks with internal dynamics). In the former case, external cascades of linear filters are used to equip a static ANN with dynamic behavior. Depending on the type of used filter we can distinguish the following models [50]:

- Nonlinear models with output feedback: nonlinear autoregressive model with exogenous input, nonlinear output error model, nonlinear autoregressive moving average model with exogenous input, nonlinear Box-Jenkins model
- Nonlinear finite impulse response model
- Nonlinear orthonormal basis function Model

Neural networks with internal dynamics are equipped with dynamic elements inside the model structure. Depending on their type we can distinguish the following networks [2]:

- Recurrent networks
- Partially recurrent networks
- Locally recurrent globally feedforward models

Presented classification certainly contains only basic models and is not comprehensive. Refer, for example, to [2, 51, 52] for more details on using ANNs for system identification. An interesting direction of contemporary research aims to application of recently flourished approach of deep neural network learning to system identification, refer to [53, 54] for more details on this topic.

1.2 Review of Bayesian Networks

A Bayesian network (BN) is a probabilistic graphical model that uses a directed acyclic graph to represent interconnections between random variables. The Bayes theorem, formulated in 1763 by the English statistician and philosopher Thomas Bayes [55] represents the central type of reasoning in these models, therefore they have got the word “Bayesian” in their name [1].

The idea of representation interactions between random variables using a directed acyclic graph originates in works of geneticist Sewal Wright. He formulated the method of path coefficients that served to analyze linear correlations between random variables. In addition to mathematical calculations, the method used graphs to interpret dependencies between multiple variables. The directed edges in a graph represented causal relationship between two corresponding variables. Wright used this method to analyze birth weight of guinea pigs and transpiration of plants [56]. This research work, which was published in 1921, is considered the first appearance of a model which we now call a Bayesian network [1]. More detailed description of S. Wright’s method and further applications may be found in his later work [57].

The idea of using a directed graph to reflect causal relationships was adopted in other disciplines. In particular, it appeared in the work of Swedish econometrist and statistician Herman Wold [58] and in the book of American sociologist Hubert Blalock Jr. [59].

At the same time, the distinguished statistical geneticist Robert C. Elston and his colleagues published their results in the research of human heredity [60, 61]. Their aim was to test specific genetic hypotheses regarding genotypes and phenotypes of individuals using the pedigree chart represented via a directed acyclic graph. On the basis of their research, they invented so called Elston-Steward algorithm for computation the likelihood of observed genotype given a pedigree.

Despite the success of mentioned applications, probabilistic models were widely rejected by statisticians for decades. The main reason was probably the substantially low acceptance of Bayesian statistics in research community at that time. The disagreement between frequentist (also called orthodox) and subjective (also called

Bayesian) view of probability was the topic of wide discussion [62]. For more details a reader may refer to further publications, for example to [63, 64] on the frequentist side and to [65, 66] on the Bayesian side.

In the area of computer science, probabilistic models found their first usage in the computer-aided medical diagnosis. The idea of using the Bayes' rule in medical diagnosis firstly appears in 1959 in the Science journal [67]. Authors suggest that assessing the probability that a patient has a certain disease given an observed set of symptoms can be calculated from the probability of appearance of these symptoms given the disease (that is the reverse to the conditional probability of our interest) and marginal probabilities of given symptoms and considered disease in the population from which a patient comes from. The logic behind this choice of mathematical technique is quite natural: the assessing of symptoms associated with a disease is the way how medical books are generally written (although, these assessments are often given by words "rare", "frequent", etc.). Moreover, authors emphasize the necessity of collecting sufficient amount of data for application of such approach and importance of constant renewing of population statistics to provide accurate diagnoses. The role of computers in this process and complicatedness of their implementation were also discussed by the authors in [68]. The idea of using the Bayes' rule in medical diagnosis was adopted by several groups of researchers. The pioneers were Homer Warner and his colleagues, who used it to diagnose congenital heart disease [69]. Their reasoning included 33 mutually exclusive diagnoses and 50 symptoms that were assumed to be conditionally independent given the disease. These restrictions correspond to the model of Naïve Bayes classifier, one of the simplest Bayesian networks. Despite all restrictions, the diagnoses obtained from the model agreed with actual diagnoses at least as often as did the diagnoses of three experienced cardiologists.

In the early 1970th the research group from the university of Leeds (departments of Surgery and Computational Science and the Electronic Computing Laboratory) conducted extensive work in application of Bayesian rule on medical diagnosis backed up by results of long-term application in a surgical unit. Their system for computer-aided diagnosis [70] was applied for 11 months to provide diagnoses in the field of acute abdominal pain [71]. It contained 35 discrete variables representing symptoms, previous history and personal information (sex, age etc.). Probabilities of diseases given all possible combinations of symptoms were early calculated from 600 medical cases [72]. On the base of the survey that included 304 patients, authors claim quite impressive results: the system succeeded to gain the correct diagnosis in 91.8%. This value was compared with the accuracy of diagnosis for different groups of clinicians.

Even the most senior clinicians who resulted with accuracy of 79.6% were defeated by the system [71]. In the next study authors also investigate reasons for occasional mistakes in clinicians' diagnoses. They stated and approved that clinicians were not good in estimation of probabilities, particularly when they assessed large series of similar data. This disadvantage of human reasoning showed itself the most in the cases of rare diagnoses, when accuracy of clinicians' estimates of probability differed significantly from real values [73].

Nonetheless, Bayesian networks fell into disfavor of artificial intelligence community. One of the main reasons for that was the strong belief that expert systems should use similar methods to those of human intelligence. Moreover, first probabilistic models used in expert systems had very strong independence assumptions and thus seemed inflexible and improper for the majority of practical applications. Furthermore, a lot of other formalisms for reasoning under uncertainty were invented at that time [1].

Decades had been passed from the first application of the Bayesian network's ancestor in 1921 up to the time when the formalism was finally formulated in late 1980th. During that period models that can be considered now as Bayesian networks (or models closely related to them) had been used by different groups of scientists who named them differently: recursive models [58, 74], causal models [59], causal graphs [75], causal probabilistic networks [76], belief networks [77], causal networks [78], influence diagrams [79], knowledge maps [80] and so on.

Apparently, the unified framework was missing. This gap was filled in 1980th by an Israeli-American computer scientist, philosopher and the laureate of A.M. Turing Award [81], Judea Pearl, who is considered to be the inventor of Bayesian networks. The name "Bayesian networks" was proposed by him in 1985 [82]. Together with colleagues, Pearl published a sequence of relevant papers ([83, 82, 77, 84, 85] and others) that proposed using of Bayesian networks for representation of the joint probability distribution over a set of random variables using directed acyclic graph that encodes dependencies and causal relations between those variables. For complete list of Judea Pearl's publications refer to his homepage [86]. In 1988, Judea Pearl published his highly recognized book "Probabilistic Reasoning in Intelligent Systems" that formulated Bayesian network framework [87].

Foundations for efficient reasoning using Bayesian networks were formulated by Lauritzen and Spiegelhalter in their key paper published in 1988 [78]. Big contribution to complementation of theoretical knowledge was also made in the context of influence diagrams (this model can be viewed as the generalization of Bayesian networks that provides, in addition to probabilistic inference, the best decision from the possible set of actions) ([79, 88, 89, 90]). Another influential publication that contributed to formalizing the field of Bayesian networks is the book “Probabilistic reasoning in Expert Systems” of American mathematician and computer scientist Richard Neapolitan [91].

Formulation of Bayesian network framework and formation of sufficient theoretical background in the field gave a big impetus to widening of Bayesian networks in 1990th. Another reason for abruptly increased interest in these models was successful implementation of Bayesian networks in practical applications, mostly in the field of medical diagnosis [1]. Research projects in this area include the Nestor system for the diagnosing of endocrine disorders [75], the MUNIN (MUscle and Nerve Inference Network) system for the diagnosing of muscle and nerve diseases [92], the QMR-DT (Quick Medical Reference, Decision Theoretic) system (probabilistic reformulation of INTERNIST-1/QMR Knowledge Base) for the diagnosing in general internal medicine [93, 94] and the Pathfinder system for assisting in the diagnosing of lymph-node diseases [95]. The last-mentioned diagnosing system was probably the most visible one. In addition to providing the most probable diagnoses given a set of observations made by a user, the Pathfinder system suggests additional tests that may serve to narrow a probability distribution over diseases and consequently increase diagnostic accuracy. During construction of the system, researchers firstly implemented rule-based (non-probabilistic) expert system that appeared to be inflexible and inappropriate for diagnosing. The second modification of Pathfinder system used probabilistic approach and had superior performance. However, similarly to the ancestors, it was based on the Naïve Bayes model with its strong independence assumptions (all symptoms were assumed to be conditionally independent given the disease). To overcome this inaccuracy in model formulation, the system was updated to full Bayesian network that allowed removing incorrect independencies. Consequently, diagnostic accuracy of Pathfinder increased and was at least as good as that of the Pathfinder expert [96]. Moreover, during solving of Pathfinder project, importance of avoiding zero probabilities for events that are very rare but still possible was proven in practice. Gained knowledge was formulated in the manner that allows its usage in the arbitrary branch of medical diagnosis [97].

Significant application of BNs beyond the area of medical diagnosis appeared within the Vista project. This application provided operators at Mission Control Center in Houston with a decision support system for monitoring of the Space Shuttle’s propulsion systems. Former display manager provided raw complex telemetry data to flight controllers who had to monitor correct functioning of propulsions and make swift actions in the case of a failure. New system aimed do decrease cognitive load on human operators by managing the complexity of information displayed to them. In addition, in the case of a problem, the system displayed a list of the most probable disorders (according to probabilities calculated from a Bayesian network) and their expected time-criticality to assist flight controllers in making time-critical high-stakes decisions under the influence of uncertainty [98, 99].

And finally, the most widely distributed application based on Bayesian networks is without a doubt Office Assistant provided with the Microsoft Office 1997. The well-known paperclip Clippy appeared within the Lumiére project. It predicted the goals and needs of a user based on the query of his/her recent actions and provided the most relevant (according to its beliefs) help information [100].

The success of the above mentioned applications considerably reduced skepticism against Bayesian networks in the statistical and the artificial intelligence scientific communities. From that time, the Bayesian networks formalism has been spreading in different scientific areas worldwide.

An increasing interest in Bayesian networks in different scientific areas during last two decades can be backed up by results from searching engines from four popular databases of scientific articles: ScienceDirect, IEEE Xplore, Scopus and Web of Science. The figure 1.2 shows the amounts of research articles about Bayesian networks included into each of these databases in every year up to 2018. These results were obtained by searching of query “Bayesian network” in all available fields of a database (data were gathered on February 14, 2019).

During last two decades Bayesian networks were successfully implemented in different fields of study. In addition to “classical” areas of implementation, like genetics, medicine and social sciences, this tool has spread to a plenty of other disciplines. Overview of the most common areas of usage and implementation guidelines for each of them may be found for example in [101].

Recent review that includes latest tendencies in application of Bayesian networks in genetics can be found in the book of R. Neapolitan [102]. These applications include, in particular, genotype analysis and discovering of epistatic and non-epistatic interactions between genes [103, 104, 105, 106] and genetic linkage analysis [107].

Bayesian networks have been widely used in medicine, for guidelines to their implementation in this area refer to [108, 109, 110, 111]. Typical applications of Bayesian networks in medicine include: diagnosis of different diseases [112, 113, 114, 115, 116, 117, 118, 119], predicting risk of diseases [120, 121, 122, 123, 124, 125], predicting specific medical outcomes [126, 127, 128, 129, 130]. Bayesian networks are used for prediction of the human immunodeficiency virus (HIV) mutations [131]. Even though HIV virus is impossible to cure yet, effective prediction of its muta-

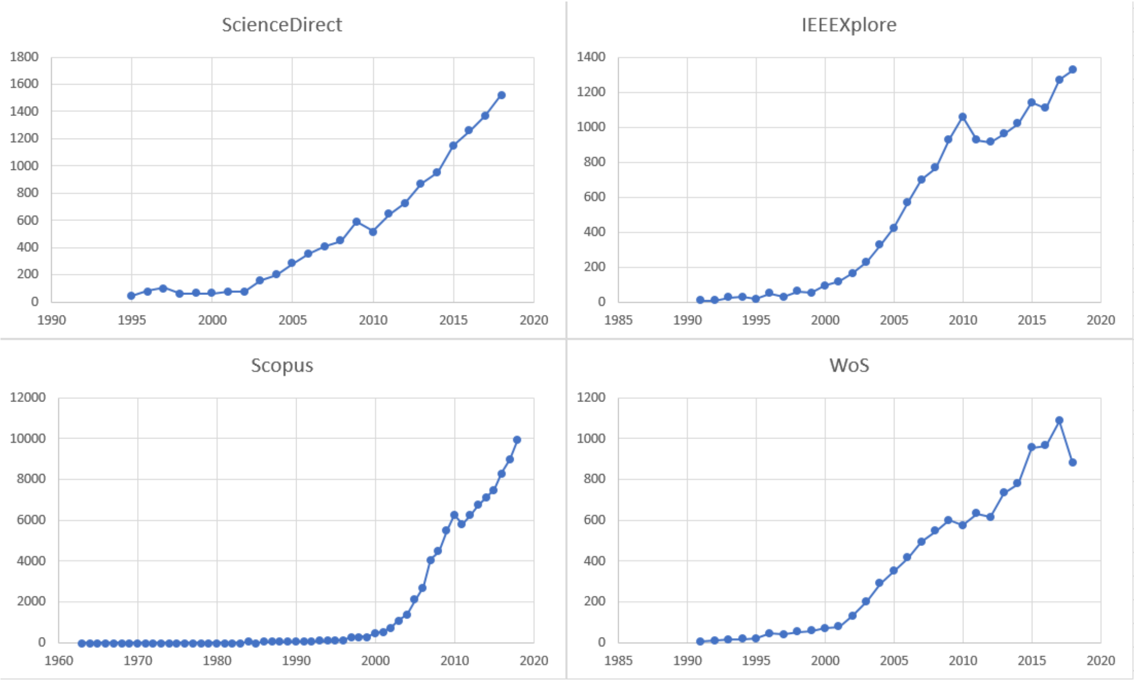


Figure 1.2: Interest in Bayesian Networks in scientific community

tions can lead to more efficient antiretroviral treatment, and therefore increase life duration and its quality for HIV-positive patients. Another promising area of application is the analysis of functional magnetic resonance imaging data of human brain activity [132, 133]. These studies have a potential to widen our knowledge about human brain functionality.

Bayesian networks can be also used to demonstrate risks of different medical interventions to lay people in comprehensible way. For example, recent paper [134] presents the medical negligence case initiated by the patient who suffered a stroke because of invasive diagnostic test. Inappropriateness of this test as compared to alternative non-invasive test was proven using Bayes theorem. However, this explanation was not clear for lay people, so researchers successfully used decision trees and Bayesian networks to explain risks of alternative scenarios to jury.

Another perspective area for Bayesian networks is forensic science. Crime investigation naturally involves uncertainties of different kind together with a lot of available statistical information from previous similar investigations. These factors create good environment for application of Bayesian networks. General guidelines on using Bayesian networks in forensic science can be found in [135]. Applications in this area include: forensic DNA identification and paternity testing [136], risk assessment of violence manifestations for prisoners with mental health problems [137, 138], crime linkage modelling [139], etc.

Bayesian networks are successfully used in environmental science [140], in particular in ecology [141, 142, 143, 144], in the research of renewable energy sources [145] and in agriculture [146, 147, 148].

In engineering Bayesian networks are used for monitoring [149], fault detection and diagnosis [150, 151, 152, 153], risk analysis [154, 155, 156, 157, 158, 159, 160, 161] and reliability assessment [162, 163, 164].

Relatively unusual, but intriguing domains for application of Bayesian networks are financial and marketing informatics [165], sport betting [166], educational assessment [167, 168], weather forecasting [169], information retrieval [170] and social network analysis [171, 172]. Reader may also come across quite unusual applications, for example the modelling of maritime piracy situation [173], teamwork improvement [174] or indoor color design [175].

The above-mentioned overview proves that Bayesian networks have approved themselves as a powerful tool for decision-making under uncertainty in different fields. Modern tendencies suggest, that this framework will be spreading to further areas with time.

1.3 Bayesian networks in control systems engineering

Since interconnection between Bayesian networks and system identification was not closely addressed in available literature and research articles, we present the state of the art from the broad perspective of control systems engineering. In some subfields (monitoring, fault detection and diagnosis) BNs have gained popularity while in others (feedback and stochastic control) they appear rarely. Since system identification methods provide these subfields with models of dynamic systems, we believe that this broad perspective will give not only the insight into the range of applications of Bayesian networks in control engineering, but also into the scale of possible applications of system identification methodology proposed in this thesis.

Bayesian networks have started to gain in popularity in the field of control engineering in the 2000s. This late appearance (in comparison with other fields) is caused mainly by the fact that it is relatively recently that BNs matured for applications in this field. By the word “maturation” we mean that essential (from the control engineering point of view) structures were formulated in their context. The most important ones are the introduction of continuous nodes into network structure and development of temporal dependencies. The former extension is required since most of variables of our interest are continuous (in that they can take a value from infinite set of values), the latter one is vital due to dynamic nature of controlled systems.

Initially Bayesian networks reasoned exclusively over discrete variables. Normally distributed variables with linear dependencies were introduced in the context of influence diagrams (graphical models that can be considered as Bayesian networks extended by decision making nodes) in 1989 [176], the framework was extended to the hybrid case (containing both continuous and discrete nodes) in 1994 [177].

The first temporal extension of Bayesian networks was proposed by Dean and Kanazawa in 1989 [178]. They called this extension dynamic Bayesian networks since they evolve in time and their current state depends on the states in previous steps. In contrary, networks that do not change over time are often referred to as static Bayesian networks. Big contribution to the development of dynamic Bayesian networks was made by Kevin Murphy. His Bayes Net Toolbox (BNT) [179, 180] for MATLAB [181] made Bayesian networks (especially the dynamic subclass) accessible for the wide community of researchers. Also, K. Murphy provided an extensive tutorial on dynamic Bayesian networks in 2002 [13]. Murphy showed, that a dynamic Bayesian network can be viewed as the generalization of Hidden Markov models and Kalman filter models and his work covers their representation, inference and learning. In addition, he provided currently the most comprehensive overview of software packages for modelling of Bayesian networks, influence diagrams and Markov networks (probabilistic graphical models described by an undirected graph) [182] that has been updated by constantly emerging packages.

It is important to mention, that the plenty of methods and research works contain the adjective “Bayesian” in their titles. However, it does not necessarily mean that they use Bayesian networks. More often, this adjective reflects the fact that a method is based either on Bayesian statistics or simply on using of Bayes’ rule. It is worth emphasizing, that Bayesian networks do not necessarily imply Bayesian statistics. In most applications the parameters of a network are considered unknown constants and classical statistical approaches (e.g. maximum likelihood estimation) are used to estimate them. But if unknown parameters have to be treated as random variables, Bayesian methods can be used in Bayesian networks. Therefore, it is important to distinguish between Bayesian methods and Bayesian networks.

For example, a term “Bayesian control” describes Bayes’ rule based control paradigm. It appears in application of stochastic models of conventional controllers [183] and in the field of statistical process control [184]. The representative of the former field is [185], where authors proposed to use Bayes’ theorem to estimate stochastic model of the inverse controller for nonlinear dynamic systems. Representation of results in the latter field requires prior introduction since the control paradigm is different from the conventional one. Statistical process control (SPC) is a method of quality control which uses statistical methods for the monitoring and control of processes. Control chart is a key tool of SPC that reflects the variation in a process [186]. It is used for tracking a variable using two statistical characteristics: a measure for centering (the mean for normally distributed variables) and a measure for spread (the standard deviation for normally distributed variables). The measure for centering defines desired value and the measure for spread defines permissible range. An equipment carries on without intervention as long as the value of variable is in a stable zone, since it is assumed that variation in signal is caused by common causes (inaccuracy of sensors, influence of noise). If signal shifts to warning zone, it is a sign that some special causes of variation may have an influence on a process and operator should consider intervention to a process. If signal moves to action zone, it alerts that something has gone wrong and intervention to process is required [186].

If design parameters of control chart (e.g. sampling parameters, control limit parameters) change over time based on values from previous time steps, the control chart is called adaptive or dynamic. There are two main streams of research in adaptive statistical process control. The first one is an extension of conventional control chart, in which sample parameters (sampling interval and sample size) can be changed dynamically whereas other parameters stay fixed. The second stream adopts Bayesian approach since the state of a process is updated in each step using Bayes' theorem. This approach is more flexible since it allows dynamical updating of the control limit parameters. Process control with such control charts is called Bayesian process control and charts themselves are also called Bayesian [184]. Bayesian control charts have a long history. Introduced in 1952 by Girshick and Rubin [187], they are still the subject of active research in the present [184, 188, 189].

Another active research area with misleading name is “Bayesian identification”. As oppose to the conventional approach, where parameters are considered unknown constants, in Bayesian methods they are treated as random variables. Consequently, we are looking for probability distributions of unknown parameters. Bayes rule is used in this context for updating of the posterior distribution based on the prior distribution of parameters and measurements obtained from a dynamic system. Basic principles of Bayesian identification are presented, for example, in [190]. These methods are the adaptation of techniques from Bayesian statistics into the domain of system identification. In addition, kernel methods for system identification are often referred to as Bayesian methods for system identification. As it was already mentioned in the section 1.1, these methods correspond to the adaptation of regularization techniques from machine learning into system identification domain, refer to [28, 29] for recent reviews in this field.

In the following text we present the state of the art in the interconnection between control systems engineering and Bayesian networks. Since we hit misleading titles quite often during its preparation, we dare to introduce the full name of sources in the cases when it is crucial to avoid misunderstanding or explain ambiguities.

The subarea of control engineering, in which Bayesian networks have become particularly popular is the monitoring of dynamic systems. Due to constant increasing of the complexity of controlled processes and growing demand for their reliability and safety, application of classical dynamic models from control theory becomes infeasible for sophisticated systems. Simplification of these models for the task of fault detection and diagnosis is provided by functional redundancy techniques that use either analytical or topological methods. Analytical methods are based on algorithms from classic control theory (filtering, state estimation, parameter estimation etc.). In topological methods relations between variables are represented by a directed graph constructed for system under normal operating conditions and in faulty modes [191]. Using of hybrid dynamic Bayesian networks in this setting was proposed by Lerner and his colleagues in 2000 [9]. He provided the introduction into hybrid dynamic Bayesian networks and carried out extensive research in this field that includes treatment of non-linearities and development of new inference algorithms [192]. His research theories are backed up by successful application of hybrid DBNs for monitoring of the Reverse Water Gas Shift System, the system designed to produce oxygen from the carbon dioxide atmosphere on Mars [5].

Bayesian networks are widely used for fault diagnosis of dynamic systems, refer to [193] for closer information on the implementation of both static and dynamic BNs for this purpose. As the representatives of data-driven approaches to fault diagnosis, BNs require the large amount of training data. In addition, prior process knowledge is essential for defining the structure of a network [15]. These and other disadvantages may be overcome by the combination of Bayesian networks with other approaches for detection and isolation of faults in dynamic processes [194]. For example, the combination of Bayesian networks with principal component analysis [10] and discriminant analysis [7] showed robustness and good performance. Combinations of data-driven and model-based approaches are referred to as hybrid approaches to fault diagnosis, the recent review that includes implementations of BN-based combinations may be found in [15].

Bayesian networks have been successfully used in monitoring of discrete event dynamic systems (DEDSs). Extensive guidelines in this area are presented in doctoral thesis “Bayesian Networks of Dynamic Systems” [4]. In the simplest case, DEDS can be represented as an automaton composed by the finite set of states of a system, the finite set of transitions between those states and the initial state of a system. In modern manufacturing DEDSs are becoming more and more complex and monitoring becomes demanding. High level of complexity leads to the intractability of these systems by classical (centralized) approaches to monitoring due to combinatorial explosions. Overcoming of these limitations can be gained by distributed approaches, which solve monitoring tasks locally for each component and combine solutions into the global decision. A distributed dynamic system is obtained by assembling of the large number of components and hence is often referred to as a network of dynamic systems. Monitoring of such networks can be provided by Bayesian networks that leads to the formulation of Bayesian networks of dynamic systems [4].

One of the most popular types of DBNs used in monitoring of dynamic systems is a Hidden Markov Model (HMM). This probabilistic graphical model was initially developed for speech and text processing as the extension of a Markov chain [195, 196] and was later adopted in monitoring and fault detection of dynamic systems [197, 198]. The state of a system described by a HMM is unobservable (hidden), however it can be “guessed” using a variable dependable on the state which can be measured. The probabilities of transitions between particular states are described by a transition model. Refer to [199] for a well-prepared introduction to HMMs. For more demanding applications this simple structure of a graphical model can be insufficient and one of HMM extensions can be used, e. g. hierarchical hidden Markov models [200].

In a HMM we consider systems with discrete hidden variables. Probabilistic graphical model that has the same graph, but consider continuous state variable is called Linear Dynamical System (LDS) [199] or linear dynamic model [201]. It is important to point out, that in the Machine learning applications this model mostly does not have exogenous input [199, 201, 202, 203, 1] and hence is used for analysis of time series [16]. Due to similarities between LDSs and HMMs, they share the same structure of inference and learning algorithms [202]. Both types of probabilistic graphical models can be represented by a dynamic Bayesian network [13] and consequently, inference and learning algorithms designed for dynamic Bayesian networks can be adopted in HMM-based and LDS-based models [204]. The model of LDS that includes input signal can be found in the thesis of K. Murphy [13]. He suggests the principle how to use this model (also referred by him as a state-space model and a Kalman filter model) for control purposes. This principle is described in details later in this section.

In the context of this discussion, it is worth to point out further ambiguity in nomenclature. It is important to distinguish between Markov models and Markov networks. A Markov network is a probabilistic graphical model represented by an undirected graph, hence it does not encode any causal relationship between random variables [1]. In a Markov model there are causal relationships between variables and hence the Bayesian network is more appropriate probabilistic graph than Markov network for this model.

An exciting property of Bayesian networks is their ability to generalize over many seemingly unrelated models and methods [205]. In addition to already mentioned HMMs and LDSs, they can also be used to implement e.g. factor analysis, principal component analysis and switching linear dynamic systems [206] (also called jump linear dynamic systems [207]). The generative model that covers all abovementioned methods and many others is called a linear Gaussian model [11]. This generative model can be transformed into a Bayesian network and analyzed using inference and learning algorithms formalized in the area of probabilistic graphical models [13]. In addition, basic models can be combined into advanced structures to solve complex tasks and this gives the extensive area of possible implementations of Bayesian networks in different engineering branches.

Applications of Bayesian networks in the control of dynamic systems are yet rare, but this direction has gotten attention of research community recently. For example, the introductory article of M. Ashcroft "An Introduction to Bayesian networks in Systems and Control" published in 2012 [208] gives an insight into the main properties of Bayesian networks and influence diagrams, into advantages and disadvantages of utilization of different node types and into inference and learning procedures. Hidden Markov models and Kalman filters are mentioned as representatives of dynamic Bayesian networks. However, the article lacks explanation on how presented models can be used in the context of systems and control. The aim of this article is to introduce basic principles from Bayesian network framework that can potentially be used in system design and control.

The main drawback of Bayesian networks in the context of process control is their disability to represent feedback, since a graph representing Bayesian networks is by its definition acyclic. More specifically, interconnections between variables in Bayesian networks are represented by a directed acyclic graph (DAG). This drawback can be overcome by introduction of feedback into the structure of a graphical model, which leads to a model called a directed cyclic graph (DCG) as proposed by Spirtes [209] or to a model called Dependency networks as proposed by Heckerman and colleagues [210]. However, this extension will have influence on representation, inference and learning of graphical models. Moreover, resulting graphs cannot be considered as Bayesian networks since one of crucial properties, the acyclicity, is not preserved.

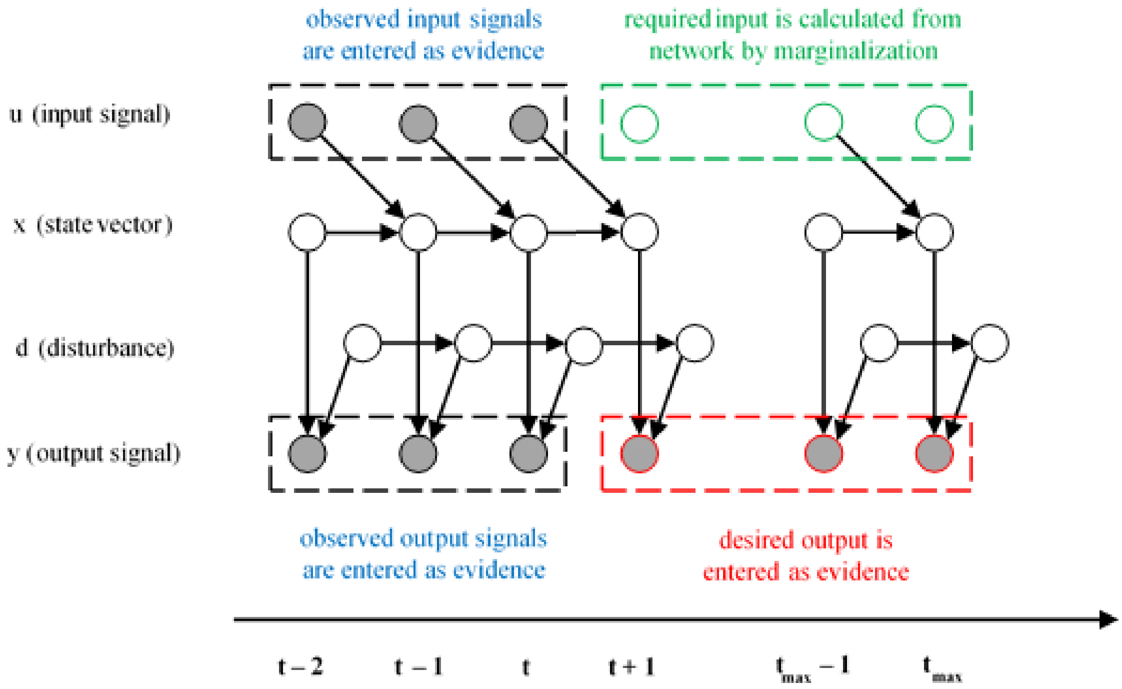

observed input signals are entered as evidence
required input is calculated from network by marginalization
u (input signal)
x (state vector)
d (disturbance)
y (output signal)
observed output signals are entered as evidence
desired output is entered as evidence
t − 2
t − 1
t
t + 1
$t_{max} - 1$
t_{max}

Figure 1.3: Control strategy proposed by R. Deventer et al.

Another way of introducing feedback into the structure of a Bayesian network is implicit implementation as proposed by Murphy [13] and implemented by Deventer and colleagues in [6]. Authors in the latter publication use a DBN-based Kalman filter model to model the behavior of a dynamic system. Implicit feedback is implemented by the special treatment of nodes in a Bayesian network as presented on the figure 1.3. Specifically, measured values of input and output variables are entered to a network as evidence for time slices up to the current time, desired values of output are entered as an evidence for future time slices. The temporal evolving of an input node is obtained by marginalization and represents the input signal for controlled system. Application of this control principle in real-time in combination with forgetting strategy (throwing away the oldest input and output values and adding the most recent ones at each time step) leads to the implementation of Bayesian adaptive controller that shows nearly the same performance as classic control methods [211]. The advantages of Bayesian controllers compared to classic control algorithms are the possibility of training the model from examples with partial knowledge regarding system structure and its parameters. Moreover, BNs offer a possibility to handle missing information that makes them superior to other self-adaptive tools (for example, neural networks) [6]. A Kalman filter is based on the state space description of dynamic systems. The resulting process fulfill Markov independence assumption since values of variables at any time step are independent of all past observations given values in the previous step. The DBN for this model hence consists of two adjacent time slices for a system of arbitrary order. An order of a system has an influence on the amount of state variables that are often unmeasurable and non-interpretable. Calculation of these variables brings additional computational burden to control process. Alternative model for representation of dynamic systems introduced by Deventer is a DBN based on a difference equation [212]. This network is represented by input and output variables and interconnections between them. In this case, there is no need to calculate state variables, but the change in model order changes structure of a DBN. Dynamic network will be represented by (n+1) time slices, where n is the order of a dynamic system. Designed control algorithms were tested on simulated data and successfully used in practice for control of hydroforming and injection moulding technological processes [12].

Models proposed in above-mentioned publications of Deventer and colleagues assume that considered dynamic systems have linear behavior. Since many controlled systems have non-linear behavior, it is important also to define a technique for including non-linearities into considered models. Bayesian networks does not put any restrictions regarding the type of interconnection between random variables. Hence, non-linear functions for representation of dependence of child nodes on parent nods can be introduced into the structure of Bayesian network. However, it is important to take into account that only linear interconnections between random variables are sufficiently implemented in available software packages. Therefore, the implementation of non-linear models requires big effort in specification of used non-linearities in the context of BNs and adaptation of inference and learning algorithms to non-linear dependencies. Another possible solution is to approximate non-linear behavior of dynamic systems by hybrid BNs as presented in [213].

Another application of a Bayesian network based controller is presented in the article of Welch and Smith [8]. Here authors apply a Bayesian network for the task of nuclear waste remediation, more concretely for the two-position control of a sorter unit. In contrary to the previous example, the model is designed using topological techniques rather than analytical approach. Complete model of technological process is presented by a relatively sophisticated hybrid Bayesian network. Since inference in full BN was not sufficiently fast for application in real time, authors used the strategy of hierarchical control [214] to reduce irrelevant variables from consideration. Reduced conditional sub-network consists of variables required for the most frequent control decisions and can be evaluated using exact methods in online setting [8].

Bayesian networks were also used for the adaptive control of a printing system [215]. The variables under consideration were divided into four groups: sensor information, control variables, hidden states and target variables. These variables were connected into a DBN structure according to the expert knowledge. Authors used DBN for prediction of the paper type based on the estimation of paper weight. This information is than used by another DBN for adaptive control of engine speed.

Based on this application authors suggest a technique for the construction of Bayesian networks for linear dynamic systems using expert knowledge regarding the process under consideration [216]. They describe how topological techniques can be used to construct diagrams of linear dynamic systems, show the transformation of these diagrams into state space representation and present the appropriate structure of a DBN. They suggest three choices for nodes in a network: deterministic, linear Gaussian and conditional linear Gaussian.

In the field of statistical process control, it was proposed to use a Bayesian network as a control chart [7]. The authors present an approach that combines both detection and diagnosing tasks in a single tool (conventional detection and diagnosing procedures use separate tools for each task). As it has been repeatedly pointed out, Bayesian networks is an approved tool for diagnosing. This article proposes solving of detection task by a control chart implemented using a Bayesian network.

Bayesian networks were also successfully used to solve trajectory optimization problems in robotics [217]. They provide an alternative interpretation of models used in motion control and planning that are particularly useful for approximate probabilistic inference in complex robotic systems.

Presented overview of applications of Bayesian networks in automatic control proves that this interconnection is not yet well defined but has a big potential for the wide range of technological processes. Trends in development and broadening of Bayesian networks suggest that this research area will flourish in the near future.

And finally, the review over available sources shows, that identification of dynamic systems was not yet closely addressed and studied in available literature. Recent overview on system identification perspectives written by Lennart Ljung backs up this statement [14]. Author suggests Bayesian networks as a perspective tool for description and calculation of signal flows in dynamic systems that has not been used in system identification.

System identification task corresponds in the Bayesian network framework to the task of parameter learning (parameter estimation) given the structure of a network and observed values of input and output variables. Some authors have already noticed this correspondence [11, 12, 13], some authors provided a basic insight on how this procedure may be implemented for linear dynamic models without external stimuli [11, 201, 199, 1, 202, 203] and with it [13]. However, the systematic study of Bayesian network based identification of dynamic systems has not been yet provided.

The only source that has drown our attention as a candidate to fill in this gap was the article “Bayesian-network-based system identification of spatial distribution of structural parameters” by S.-H. Lee and J. Song published in 2016 [218]. Authors claim that they provided the first attempt of Bayesian-network-based system identification. Their research originates from the field of structural engineering. Prevention of losses of properties and human lives due to deterioration and damage in civil infrastructures is a crucial task to solve in this field. Implementation of damage monitoring and detection is referred to as Structural Health Monitoring (SHM). System identification is defined in this context as one of the SHM methods used for estimation of structural parameters. It is defined as a method that minimizes the error between measured response of a structure and response calculated from the assumed model under the same loading conditions. In the article authors use measurements from sensors located on the structure. The input data for analysis are organized as the mesh of a given size. Authors propose to use a Bayesian network that contains measured and calculated displacements for each point in a mesh. For simplification, continuous values of displacement are discretized and a resulting Bayesian network is discrete. To make analysis more tractable, they suggest to analyze spatial distribution of structural parameters (in the form of a bivariate Gaussian) instead of particular values of displacements. By identification authors mean parameter estimation for spatial distribution of structural parameters based on the nodes in a designed Bayesian network. The another example of the implementation of Bayesian networks for identification of structural parameters is presented in the thesis of T. B. Tran “A Bayesian Network framework for probabilistic identification of model parameters from normal and accelerated tests: application to chloride ingress into concrete” defended in 2015 [219]. Here, a discrete Bayesian network is constructed from chloride contents at different depth and time and structural parameters of interest represented as parent nodes. The number of parameters to be identified depends on a selected model. Both publications are dedicated to estimation of material properties in structural engineering [220], the task that is not connected with the system identification procedure used in the context of control systems design.

1.4 Research objectives

After extensive literature review the following research objectives were chosen:

- Design Bayesian network based models of dynamic systems that can be used for modelling and system identification.
- Propose a methodology of Bayesian network based system identification
- Explore the efficiency and precision of the proposed approach using simulated responses of the most popular types of dynamic systems.
- Verify the proposed approach to system identification on real dynamic systems and compare results with traditional methods

2 Mathematical models of dynamic systems

Solving tasks from control systems engineering often requires knowing of the mathematical model of a dynamic system of our interest. These models can be found analytically, but this approach is complicated and time-consuming, especially for complex real-world technological processes. More often the models are obtained using system identification methods which were briefly discussed in the section 1.1. These methods often assume that the type and the structure of a model for a studied system are known. The matching of real-world process with a structure that describes their behavior with sufficient precision may be a challenging issue. Choosing of appropriate model often involves a trade-off between precision and complexity in the context of the task for which system identification is performed. The oversimplified model can provide conclusions that are not valid in the real world, whereas the overcomplicated model can bring unnecessary computational difficulties.

Bridging between the real world and mathematical theory can be provided by the classification of different dynamic systems according to their properties and the introduction of corresponding mathematical models which describe their behavior. This chapter provides the description of major classes of dynamic systems (section 2.1) and introduces mathematical models used for subclasses considered in this thesis (sections 2.2 - 2.3). For more detailed description refer, for example, to [221]. State observers are briefly discussed in the section 2.4.

2.1 Classification of dynamic systems

From the viewpoint of control theory mathematical modelling of dynamic systems can be provided in frequency or time domain. System identification using the former approach is time-consuming compared to the latter one, but it is however irreplaceable for systems which demand knowing of system response on the harmonic signal in a wide frequency spectrum. These methods and mathematical models associated with them are not in the scope of this thesis. A reader may refer to any of the following books for more information on this approach to system identification [16, 17, 2, 26].

Mathematical models used for analysis of dynamic systems in time domain describe relations between an input signal and an output signal by an equation. The type of this equation depends on the type of a considered system. System identification for these systems mostly consists in searching for unknown parameters of pre-defined equation. A powerful extension of this modelling paradigm is a state space representation, in which we additionally consider the values of internal variables (called states) of a dynamic system. The original equation is transformed into the system of equations of the first order, which can be easily adapted for systems with several inputs and/or several outputs.

The major classes of systems are shown on the figure 2.1. Each division is based on a distinct feature of a system, hence, each dynamic system can belong to several different classes [221].

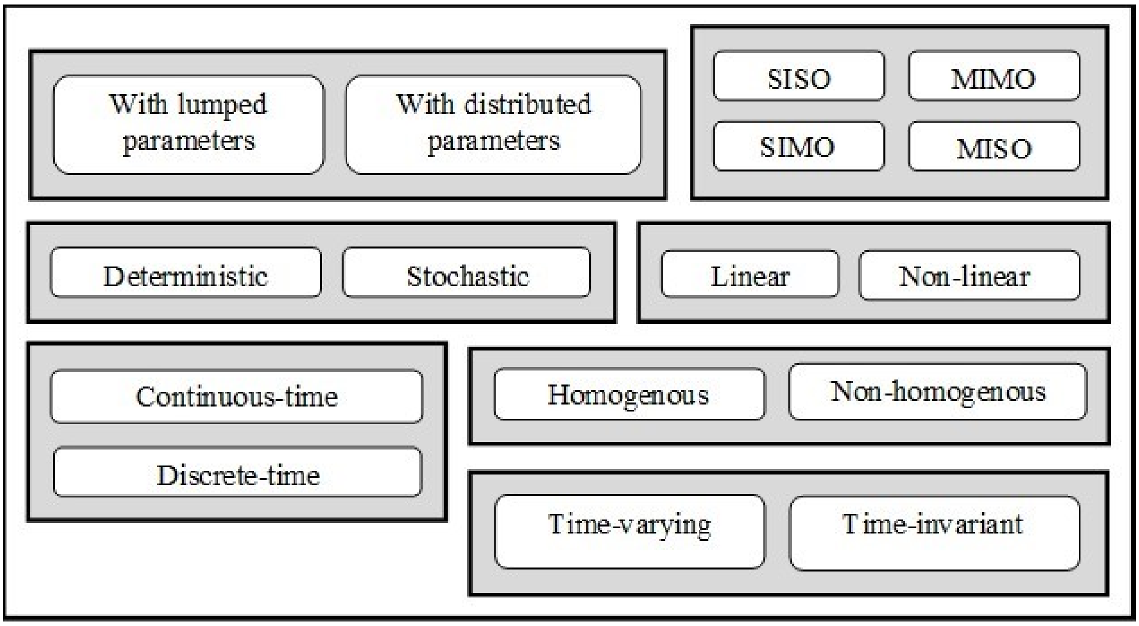


Figure 2.1: The major classes of dynamic systems

In **continuous-time** systems all signals are functions of time, in **discrete-time** systems signals are represented via sequences of values at distinct time instances with a sample period defined for each system individually. The majority of dynamic systems has continuous nature, but the majority of modern control systems are discrete. Hence there are two basic principles for coping with this issue: to consider all control loop as a discrete or to use converters for analog/discrete transformations of signals. Mathematical models for continuous dynamic systems are based on a wide range of differential equations. Discrete systems are described by difference equations with forward or backward differences.

Non-linear dynamic systems are systems in which the principles of superposition do not apply. **Linear** dynamic systems are more of idealization of the real world since most of systems exhibit non-linear behavior in practice (except the simplest ones). However, many of them can be treated as linear systems, at least for a range of input values near an operating point. Linear dynamic systems are analyzed using a linear equation (differential or difference), whereas non-linear systems include non-linear functions or elements in their description.

Depending on the amount of input and output signals, dynamic systems are divided into four categories: **SISO** (single input single output), **MIMO** (multi input multi output), **SIMO** (single input multi output) and **MISO** (multi input single output). The amount of input and output signals influences the amount of equations required to fully describe the behavior of a considered system.

A dynamic system without external input is called a **homogeneous** system. The behavior of such system is defined solely by the values of its initial conditions and is described by a homogeneous equation. A system that does have external inputs that influence its behavior is called **non-homogenous**.

Systems **with distributed parameters** are systems with infinite-dimensional state space. The typical representative for mathematical description of such systems is a partial differential equation. Their counterpart, the systems **with lumped parameters**, can be analyzed using ordinary differential equations.

Mathematical models that respect the influence of random components on the behavior of dynamic systems are called **stochastic**. In contrary, the output from **deterministic** systems is unequivocally determined by input signals and the values of output signal in previous time steps with no randomness engaged.

In **time-varying** systems parameters change over time, while in **time-invariant** systems they remain constant.

One additional class of dynamic systems that did not appear in the classification on the figure 2.1, but it is worth mentioning in the context of this discussion, is the class of **hybrid** dynamic systems. The behavior of hybrid systems is determined by interacting discrete and continuous dynamics. Mathematical models for this type of systems include both continuous and discrete variables. The most popular subclass of hybrid systems is switching affine models defined as a collection of linear (affine) models enriched by the so-called discrete state (the additional discrete variable which is used for switching between individual affine models). In the simplest case, this state is a deterministic variable with the finite set of values and the resulting system is called a jump linear model. In more complex case, the discrete state is determined by a polyhedral partition of the state-input domain. The resulting system, that can approximate nearly any non-linear dynamics, is called piecewise affine model. For an extend tutorial into subclasses of hybrid systems and their identification refer to [222].

Dynamic systems considered in this thesis correspond to the subclass of linear time-invariant non-homogenous SISO dynamic systems with lumped parameters. In further text we will let off all epithets for simplification and use simply a “dynamic system”, unless it is important to name some of them.

Even though the considered subclass may appear limited, it exhibits sufficient accuracy for many industrial processes [19, 30]. Moreover, the extensions of proposed approaches, which are presented in the final chapter of this thesis (section 7.3), will hopefully allow to use Bayesian networks in a wide range of dynamic systems in the future.

Bayesian network based models of dynamic systems considered in this thesis correspond to discrete representation. As it was already mentioned, the majority of dynamic systems have continuous nature, the discrete description appears by sampling during their usage with discrete circuits (e.g. measurement or control units). Both types of description are often used in practice, since each of them has benefits in different tasks. For example, it may be more convenient to sample continuous models during simulations with different discretization steps since the coefficients of a mathematical model do not change for various discretization steps (unlike the discrete model). On the other hand, including of a stochastic component into a description of discrete systems is more straightforward.

The description of both types of dynamic systems is presented in the following sections. An overview of mathematical models used for description of dynamic systems in continuous-time domain is provided in the section 2.2. The section 2.3 is dedicated to the discrete-time models.

2.2 Continuous-time dynamic systems

The behaviour of a continuous-time dynamic system can be in general described by the following differential equation:

$$\begin{gathered}\frac{d^n y(t)}{dt^n} + a_{n-1} \cdot \frac{d^{n-1} y(t)}{dt^{n-1}} + ... + a_1 \cdot \frac{dy(t)}{dt} + a_0 \cdot y(t) = \\ b_m \cdot \frac{d^m u(t)}{dt^m} + b_{m-1} \cdot \frac{d^{m-1} u(t)}{dt^{m-1}} + ... + b_1 \cdot \frac{du(t)}{dt} + b_0 \cdot u(t).\end{gathered} \tag{2.1}$$

In the equation (2.1), $y(t)$ is an output signal, $u(t)$ is an input signal, n is the order of a dynamic system. The condition $m \leq n$ has to be fulfilled for a considered system to be causal.

Solving of the differential equation (2.1) requires knowing of the following initial conditions (given that t_0 is initial time):

$$\begin{aligned} &\frac{d^{n-1}y(t_0)}{dt^{n-1}}, \frac{d^{n-2}y(t_0)}{dt^{n-2}}, \cdots, \frac{dy(t_0)}{dt}, y(t_0); \\ &\frac{d^{m-1}u(t_0)}{dt^{m-1}}, \frac{d^{m-2}u(t_0)}{dt^{m-2}}, \cdots, \frac{du(t_0)}{dt}, u(t_0). \end{aligned} \tag{2.2}$$

A transfer function is the ratio of the Laplace transform of the output to the Laplace transform of the input with the assumption that all of the initial conditions on the system (2.2) are zero:

$$G(s) = \frac{Y(s)}{U(s)}. \tag{2.3}$$

The Laplace transform of the output signal can be found as:

$$Y(s) = \mathcal{L}\{y(t)\} = \int_0^\infty y(t) \cdot e^{-st} dt. \tag{2.4}$$

The Laplace transform of the input signal can be found as:

$$U(s) = \mathcal{L}\{u(t)\} = \int_0^\infty u(t) \cdot e^{-st} dt. \tag{2.5}$$

Applying the Laplace transform on both sides of the differential equation (2.1) and respecting the ratio (2.3), we obtain the transfer function in the following form:

$$G(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0}. \tag{2.6}$$

The state space representation of dynamic systems is a mathematical model that describes interconnections between input variables, state variables and output variables. State variables are internal relatively to an investigated dynamic process. These variables can but do not have to reflect some physical values in the dynamic process. The state space model is based on the reorganization of differential (or difference for discrete systems) equation of a system into a system of the differential equations of the first order.

A state space representation is not unique for a system, because it depends on the choice of the state vector. However, all state space models that reflect the behavior of a certain system can be transformed to unique description (e.g. transfer function or differential/difference equation).

State space description of SISO continuous dynamic systems can be defined as:

$$\begin{aligned} \frac{d\mathbb{X}(t)}{dt} &= \mathbb{A}\cdot\mathbb{X}(t)+\mathbb{B}\cdot u(t),\\ y(t) &= \mathbb{C}\cdot\mathbb{X}(t)+\mathbb{D}\cdot u(t). \end{aligned} \tag{2.7}$$

In the equation (2.7), $\mathbb{A}\in\mathbb{R}^{n\times n}$, $\mathbb{B}\in\mathbb{R}^{n\times 1}$, $\mathbb{C}\in\mathbb{R}^{1\times n}$ and $\mathbb{D}\in\mathbb{R}^{1\times 1}$ are state matrices, $\mathbb{X}(t)\in\mathbb{R}^{n\times 1}$ is a state vector:

$$\mathbb{X}(t)=\begin{bmatrix} x_1(t) & x_2(t) & \cdots & x_n(t)\end{bmatrix}^T. \tag{2.8}$$

The main advantage of state space representation compared to other descriptions of dynamic systems is its generalizability over SISO and MIMO systems. Description of MIMO systems using differential equations (2.1) or transfer functions (2.3) requires definition of one equation for each pair of input and output signal. The state space model equations (2.7) remain the same for the arbitrary amount of input and output signals, but scalar input and output variables have to be exchanged by input and output vectors and the dimensions of state matrices have to be modified accordingly.

The solution of the first equation in (2.7) is:

$$\mathbb{X}(t)=e^{\mathbb{A}(t-t_0)}\cdot\mathbb{X}(t_0)+\mathbb{B}\cdot\int_{t_0}^{t}e^{\mathbb{A}(t-\tau)}u(\tau)d\tau. \tag{2.9}$$

In the equation (2.9), $\mathbb{X}(t_0)$ is the vector of initial conditions for state variables. If initial conditions (2.2) are equal to zero, then the initial values of states are equal to zero as well. Otherwise, their values have to be recalculated using the following formula:

$$\mathbb{X}(t_0)=\mathbb{P}^{-1}(\mathbb{Y}(t_0)-\mathbb{S}\mathbb{U}(t_0)). \tag{2.10}$$

In the equation (2.10) $\mathbb{Y}(t_0)$ is the vector of initial conditions for the output signal and $\mathbb{U}(t_0)$ is the vector of initial conditions for the input signal:

$$\mathbb{Y}(t_0)=\begin{bmatrix} y(t_0)\\ \frac{dy(t_0)}{dt}\\ \cdots\\ \frac{d^{n-1}y(t_0)}{dt^{n-1}}\end{bmatrix},\quad \mathbb{U}(t_0)=\begin{bmatrix} u(t_0)\\ \frac{du(t_0)}{dt}\\ \cdots\\ \frac{d^{n-1}u(t_0)}{dt^{n-1}}\end{bmatrix}. \tag{2.11}$$

Matrices $\mathbb{P}$ and $\mathbb{S}$ are composed from the state matrices:

$$\mathbb{P} = \begin{bmatrix} \mathbb{C} \\ \mathbb{CA} \\ \cdots \\ \mathbb{CA}^{n-1} \end{bmatrix}, \quad \mathbb{S} = \begin{bmatrix} \mathbb{D} & 0 & 0 & \cdots & 0 \\ \mathbb{CB} & \mathbb{D} & 0 & \cdots & 0 \\ \mathbb{CAB} & \mathbb{CB} & \mathbb{D} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \mathbb{CA}^{n-2}\mathbb{B} & \mathbb{CA}^{n-3}\mathbb{B} & \mathbb{CA}^{n-4}\mathbb{B} & \cdots & \mathbb{D} \end{bmatrix}. \tag{2.12}$$

As it was already mentioned above, matrices of the state space representation depend on the choice of the state vector. There are several special choices of the state vector that lead to the so-called canonical forms of the state space representation. We will consider two of them: the controllable canonical and the observable canonical.

The controllable canonical form of the state space representation is represented using the following state matrices:

$$\begin{gathered} \mathbb{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix}, \quad \mathbb{B} = \begin{bmatrix} 0 \\ 0 \\ \cdots \\ 0 \\ 1 \end{bmatrix}, \\ \mathbb{C} = \begin{bmatrix} (b_0 - b_n a_0) & (b_1 - b_n a_1) & \cdots & (b_{n-1} - b_n a_{n-1}) \end{bmatrix}, \\ \mathbb{D} = \begin{bmatrix} b_n \end{bmatrix}. \end{gathered} \tag{2.13}$$

The observable canonical form of the state space representation is represented using the following state matrices:

$$\begin{gathered} \mathbb{A} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -a_0 \\ 1 & 0 & \cdots & 0 & -a_1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & a_{n-2} \\ 0 & 0 & \cdots & 1 & -a_{n-1} \end{bmatrix}, \quad \mathbb{B} = \begin{bmatrix} (b_0 - b_n a_0) \\ (b_1 - b_n a_1) \\ \cdots \\ (b_{n-2} - b_n a_{n-2}) \\ (b_{n-1} - b_n a_{n-1}) \end{bmatrix}, \\ \mathbb{C} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix}, \qquad \mathbb{D} = \begin{bmatrix} b_n \end{bmatrix}. \end{gathered} \tag{2.14}$$

The parameters of state matrices in both equation (2.13) and equation (2.14) are the coefficients of the original differential equation (2.1), therefore both canonical representations provide unequivocal descriptions for a dynamic system. Unlike the canonical forms, general state space model is non-unequivocal, however it can be transformed into an unequivocal description, namely the transfer function, using the following formula:

$$G(s) = \mathbb{C}(s\mathbb{I} - \mathbb{A})^{-1}\mathbb{B} + \mathbb{D}. \tag{2.15}$$

In the equation (2.15), $\mathbb{I}$ is a unitary matrix. The transfer function (2.15) can be consequently transformed into the differential equation (2.1) using inverse Laplace transform.

2.3 Discrete-time dynamic systems

The behaviour of discrete-time dynamic systems can be described by a difference equation. Depending on the task, for which a model is constructed, a difference equation with positive or negative shift can be used. In general, difference equation with positive shift used for description of linear dynamic systems can be written as:

$$\begin{gathered} y(k+n) + a_{n-1} \cdot y(k+n-1) + \cdots + a_1 \cdot y(k+1) + a_0 \cdot y(k) = \\ b_m \cdot u(k+m) + b_{m-1} \cdot u(k+m-1) + \cdots + b_1 \cdot u(k+1) + b_0 \cdot u(k). \end{gathered} \tag{2.16}$$

Difference equation with negative shift can be in general written as:

$$\begin{gathered} y(k) + a_{n-1} \cdot y(k-1) + \cdots + a_1 \cdot y(k-n+1) + a_0 \cdot y(k-n) = \\ b_m \cdot u(k) + b_{m-1} \cdot u(k-1) + \cdots + b_1 \cdot u(k-m+1) + b_0 \cdot u(k-m). \end{gathered} \tag{2.17}$$

Here it is important to point out, that the coefficients in the difference equations (2.16) and (2.17) will be the same for the same dynamic system. However, if we discretize the differential equation (2.1), these coefficients will differ from those from the continuous-time description. Moreover, their values are dependable on the choice of a sampling period.

The initial conditions for discrete-time systems can be formulated as (given that t_0 is initial time instance):

$$\begin{gathered} y(t_0-1), y(t_0-2), ..., y(t_0-n); \\ u(t_0-1), u(t_0-2), ..., u(t_0-m). \end{gathered} \tag{2.18}$$

A discrete transfer function is the ratio of the Z-transform of the output to the Z-transform of the input with the assumption that all of the initial conditions on the system (2.18) are zero:

$$G(z) = \frac{Y(z)}{U(z)}. \tag{2.19}$$

The Z-transform of the output signal is defined as:

$$Y(z) = \mathcal{Z}\{y(t)\} = \mathcal{Z}\{y(kT)\} = \sum_{k=0}^{\infty} y(kT) \cdot z^{-k}. \tag{2.20}$$

The Z-transform of the input signal is defined as:

$$U(z) = \mathcal{Z}\{u(t)\} = \mathcal{Z}\{u(kT)\} = \sum_{k=0}^{\infty} u(kT) \cdot z^{-k}. \tag{2.21}$$

In the equations (2.20) and (2.21) k takes zero or positive integers and T is the sampling period. It is common to drop the sampling period from notions of the output and the input signal and denote them $y(k)$ and $u(k)$ respectively. We will use this denotation further in the thesis.

The discrete transfer function can be written as:

$$\begin{gathered} G(z) = \frac{b_m + b_{m-1}z^{-1} + \cdots + b_1 z^{-m+1} + b_0 z^{-m}}{1 + a_{n-1}z^{-1} + \cdots + a_1 z^{-n+1} + a_0 z^{-n}} = \\ \frac{b_m z^m + b_{m-1}z^{m-1} + \cdots + b_1 z + b_0}{s^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0}. \end{gathered} \tag{2.22}$$

The discrete state space representation can be defined as:

$$\begin{aligned} \mathbb{X}(k+1) &= \mathbb{A} \cdot \mathbb{X}(k) + \mathbb{B} \cdot u(k), \\ y(k) &= \mathbb{C} \cdot \mathbb{X}(k) + \mathbb{D} \cdot u(k). \end{aligned} \tag{2.23}$$

The dimensions of state matrices are for discrete-time systems the same as for continuous-time systems, and the implementation of MIMO systems also requires only modification of these dimensions, the model itself does not change. The state vector for discrete state space representation has the following form:

$$\mathbb{X}(k) = \begin{bmatrix} x_1(k) & x_2(k) & \cdots & x_n(k) \end{bmatrix}^T. \tag{2.24}$$

The state matrices of a system will not be the same in continuous-time and discrete-time representations. The transformation of the continuous state space model with state matrices $\mathbb{A}_c$, $\mathbb{B}_c$, $\mathbb{C}_c$, $\mathbb{D}_c$ into the discrete state space model with state matrices $\mathbb{A}_d$, $\mathbb{B}_d$, $\mathbb{C}_d$, $\mathbb{D}_d$ can be provided as:

$$\begin{aligned} \mathbb{A}_d &= e^{\mathbb{A}_c T}, & \mathbb{B}_d &= \mathbb{B}_c \cdot \textstyle\int_0^T e^{\mathbb{A}_c \tau} d\tau, \\ \mathbb{C}_d &= \mathbb{C}_c, & \mathbb{D}_d &= \mathbb{D}_c. \end{aligned} \tag{2.25}$$

The solution of the first equation in (2.23) is:

$$\mathbb{X}(k) = \mathbb{A}^k \cdot \mathbb{X}(t_0) + \sum_{j=1}^{k-1} \mathbb{A}^{k-1-j}\mathbb{B} \cdot u(j). \tag{2.26}$$

The vector of initial conditions $\mathbb{X}(t_0)$ contains zero values, if the initial conditions for input and output signals (2.18) are equal to zero. Otherwise, they can be recalculated using the formula (2.10). Matrices $\mathbb{P}$ and $\mathbb{S}$ have the same form as in the continuous-time case (refer to (2.12)), the vectors of initial conditions for the output and the input signal have to be adopted for discrete-time description:

$$\mathbb{Y}(t_0) = \begin{bmatrix} y(t_0 - n) \\ y(t_0 - n + 1) \\ \dots \\ y(t_0 - 1) \end{bmatrix}, \quad \mathbb{U}(t_0) = \begin{bmatrix} u(t_0 - n) \\ u(t_0 - n + 1) \\ \dots \\ u(t_0 - 1) \end{bmatrix}. \tag{2.27}$$

The controllable canonical form of the state space representation is described by the state matrices (2.13), the observable canonical form is described by the state matrices (2.14).

The transformation of state space representation to the discrete transfer function can be provided using the following formula:

$$G(z) = \mathbb{C}(z\mathbb{I} - \mathbb{A})^{-1}\mathbb{B} + \mathbb{D}. \tag{2.28}$$

2.4 State observers

State observers estimate the values of state variables in cases when they are immeasurable, but required for certain purposes, for example control or monitoring. The most natural and easy way is to estimate the values of state vector using input signal and state matrices obtained from the system identification procedure (or mathematical modelling):

$$\hat{\mathbb{X}}(k+1) = \mathbb{A}_E \cdot \hat{\mathbb{X}}(k) + \mathbb{B}_E \cdot u(k). \tag{2.29}$$

In the equation (2.29) $\mathbb{A}_E$ and $\mathbb{B}_E$ are estimated state matrices, $u(k)$ is input signal and $\hat{\mathbb{X}}(k)$ is estimated state vector. This model of state observer does not take into account neither stochastic nature of considered signals, nor the imperfectness of state matrices obtained from the identification procedure. In addition, the improper initial values of the state vector negatively influence the precision of estimated state variables.

Estimation error can be defined as:

$$\begin{gathered} \Delta\mathbb{X}(k+1) = \mathbb{X}(k+1) - \hat{\mathbb{X}}(k+1) = \\ \mathbb{A} \cdot \mathbb{X}(k) - \mathbb{A}_E \cdot \hat{\mathbb{X}}(k) + (\mathbb{B} - \mathbb{B}_E) \cdot u(k). \end{gathered} \tag{2.30}$$

Let us assume that the state matrices obtained from the system identification procedure are correct, i.e. $\mathbb{A}_E = \mathbb{A}$ and $\mathbb{B}_E = \mathbb{B}$. Than the estimation error (2.30) simplifies to:

$$\Delta\mathbb{X}(k+1) = \mathbb{A} \cdot (\mathbb{X}(k) - \hat{\mathbb{X}}(k)) = \mathbb{A} \cdot \Delta\mathbb{X}(k). \tag{2.31}$$

If $\Delta\mathbb{X}(k) = \mathbb{X}(k) - \hat{\mathbb{X}}(k) = 0$, then state estimator works properly. However, if initial values of estimated state vector are not equal to the real values of state vector, then the estimation error has dynamic behavior described by a matrix $\mathbb{A}$. If the eigenvalues of this matrix lie inside the unity circle (which is always the case if the identification was successful), then the steady state of estimation error is equal to zero and the speed of its convergence to zero value depends only on a matrix $\mathbb{A}$. If we want to accelerate the convergence of estimation error, take into account the stochastic nature of considered dynamic system and changes in system behavior, we have to modify the structure of state observer presented in the equation (2.29). This leads to different state observers formulated to meet various requirements on state estimation. Recent overview and the classification of different state observers can be found, for example, in [223].

The simplest type of state observer is Luenberger observer. It accelerates the convergence of estimation error by adding additional information about the quality of estimates. This information is obtained from the difference between measured and estimated value of the output:

$$\Delta y(k) = y(k) - \hat{y}(k). \tag{2.32}$$

This difference (multiplied by a constant) is added to the equation (2.29). Respecting the dimensions of state vector, we have to multiply it by a vector of constants that is commonly denoted as $\mathbb{L}$ and called Luenberger observer gain:

$$\hat{\mathbb{X}}(k+1) = \mathbb{A}_E \cdot \hat{\mathbb{X}}(k) + \mathbb{B}_E \cdot u(k) + \mathbb{L} \cdot \Delta y(k). \tag{2.33}$$

The estimation error will be in this setting equal to:

$$\begin{gathered} \Delta\mathbb{X}(k+1) = \mathbb{X}(k+1) - \hat{\mathbb{X}}(k+1) = \\ \mathbb{A} \cdot \mathbb{X}(k) - \mathbb{A}_E \cdot \hat{\mathbb{X}}(k) + (\mathbb{B} - \mathbb{B}_E) \cdot u(k) - \mathbb{L} \cdot \mathbb{C} \cdot (\mathbb{X}(k) - \hat{\mathbb{X}}(k)). \end{gathered} \tag{2.34}$$

If $\mathbb{A}_E = \mathbb{A}$ and $\mathbb{B}_E = \mathbb{B}$, then the estimation error simplifies to:

$$\Delta\mathbb{X}(k+1) = (\mathbb{A} - \mathbb{L} \cdot \mathbb{C}) \cdot \Delta\mathbb{X}(k). \tag{2.35}$$

Dynamic properties of the estimation error are therefore given by the eigenvalues of matrix $(\mathbb{A} - \mathbb{L} \cdot \mathbb{C})$. Hence, they can be changed with respect to requirements by choosing the appropriate values in vector $\mathbb{L}$.

Another well-known and often used type of a state observer is Kalman filter (refer to the section 1.1 for more information). According to [223], all Kalman filter based state observers (e.g. UKF, EKF) are representatives of so-called Bayesian estimators, since they respect the stochastic nature of considered signals and they are based on the estimation of the probability distribution of state variables. This is another example of misleading name, since these estimators are not connected with Bayesian networks. However, in the section 4.3 we propose, how this connection can be made.

Kalman filter uses the following model of the stochastic discrete state space representation:

$$\begin{aligned} \mathbb{X}(k+1) &= \mathbb{A} \cdot \mathbb{X}(k) + \mathbb{B} \cdot u(k) + \mathbb{V} \cdot \nu(k), \\ y(k) &= \mathbb{C} \cdot \mathbb{X}(k) + n(k). \end{aligned} \tag{2.36}$$

In the equation (2.36), $\mathbb{A}$,$\mathbb{B}$ and $\mathbb{C}$ are state matrices, $\nu(k)$ and $n(k)$ are white noise signals and $\mathbb{V}$ describes the influence of white noise on each state variable.

The model presented in the equation (2.36) does not have direct feedthrough. It corresponds to a traditional Kalman filter as presented by Kalman [39]. There are modifications of Kalman filter that include matrix $\mathbb{D}$ into the structure of underlying state space representation, but they will not be discussed in this thesis.

In addition to the above mentioned parameters we have to specify matrices $\mathbb{M}$ and $\mathbb{N}$. They are covariance matrices of noise signals $\nu(k)$ and $n(k)$ that are assumed to be uncorrelated with state vector (both real $\mathbb{X}(k)$ and estimated $\hat{\mathbb{X}}(k)$):

$$\begin{aligned} \mathbb{M} &= E\{\nu(k) \cdot \nu(k)^T\}, \\ \mathbb{N} &= E\{n(k) \cdot n(k)^T\}. \end{aligned} \tag{2.37}$$

Matrix $\mathbb{P}$ is defined as the covariance of states:

$$\mathbb{P}(k) = E\{(\hat{\mathbb{X}}(k)) - \mathbb{X}(k)) \cdot (\hat{\mathbb{X}}(k)) - \mathbb{X}(k))^T\}. \tag{2.38}$$

The derivation of the algorithm for state estimation using Kalman filter is omitted, it can be found for example in [2]. The resulting algorithm includes two main steps: prediction of state vector and correction of the predicted value with respect to the measured value of the output:

$$\begin{gathered}
\textit{Prediction step}\\
\hat{\mathbb{X}}(k+1|k) = \mathbb{A}\hat{\mathbb{X}}(k) + \mathbb{B}u(k),\\
\mathbb{P}^{-}(k+1) = \mathbb{A}\mathbb{P}(k)\mathbb{A}^{T} + \mathbb{V}\mathbb{M}\mathbb{V}^{T}.\\
\textit{Correction step}\\
\mathbb{K}(k+1) = \mathbb{P}^{-}(k+1)\mathbb{C}^{T}\cdot(\mathbb{C}\mathbb{P}^{-}(k+1)\mathbb{C}^{T} + \mathbb{N})^{-1},\\
\hat{\mathbb{X}}(k+1|k+1) = \hat{\mathbb{X}}(k+1|k) + \mathbb{K}(k+1)\cdot(y(k+1) - \mathbb{C}\hat{\mathbb{X}}(k+1|k)),\\
\mathbb{P}(k+1) = (\mathbb{I} - \mathbb{K}(k+1)\mathbb{C})\cdot\mathbb{P}^{-}(k+1).
\end{gathered} \tag{2.39}$$

The main drawback of algorithm presented in (2.39) is high computational expense, since matrices $\mathbb{K}(k)$ and $\mathbb{P}(k)$ has to be calculated at each step. The steady-state Kalman filter is the simplification of traditional Kalman filter that can be used for linear time-invariant dynamic systems. It was proven that for this subclass of dynamic systems both $\mathbb{K}(k)$ and $\mathbb{P}(k)$ tend to constant values as $k \to \infty$. Steady-state values of these parameters $\bar{\mathbb{K}}(k)$ and $\bar{\mathbb{P}}(k)$) can be calculated in advance and then be used for filtering. The algorithm reduces to the following:

$$\begin{gathered}
\textit{Prediction step}\\
\hat{\mathbb{X}}(k+1|k) = \mathbb{A}\hat{\mathbb{X}}(k) + \mathbb{B}u(k).\\
\textit{Correction step}\\
\hat{\mathbb{X}}(k+1|k+1) = \hat{\mathbb{X}}(k+1|k) + \bar{\mathbb{K}}(k+1)\cdot(y(k+1) - \mathbb{C}\hat{\mathbb{X}}(k+1|k)).
\end{gathered} \tag{2.40}$$

This algorithm is considerably less computationally expensive, however computation of parameters $\bar{\mathbb{K}}(k)$ and $\bar{\mathbb{P}}(k)$) brings its own challenges. In particular, computation of the steady-state value of $\bar{\mathbb{P}}(k)$ is challenging, since it requires solving of Riccatti equation. The steady-state gain of Kalman filter $\bar{\mathbb{K}}(k)$ is then calculated using the following formula:

$$\bar{\mathbb{K}} = \bar{\mathbb{P}}^{-}\mathbb{C}^{T}\cdot(\mathbb{C}\bar{\mathbb{P}}^{-}\mathbb{C}^{T} + \mathbb{N})^{-1}. \tag{2.41}$$

3 Bayesian networks

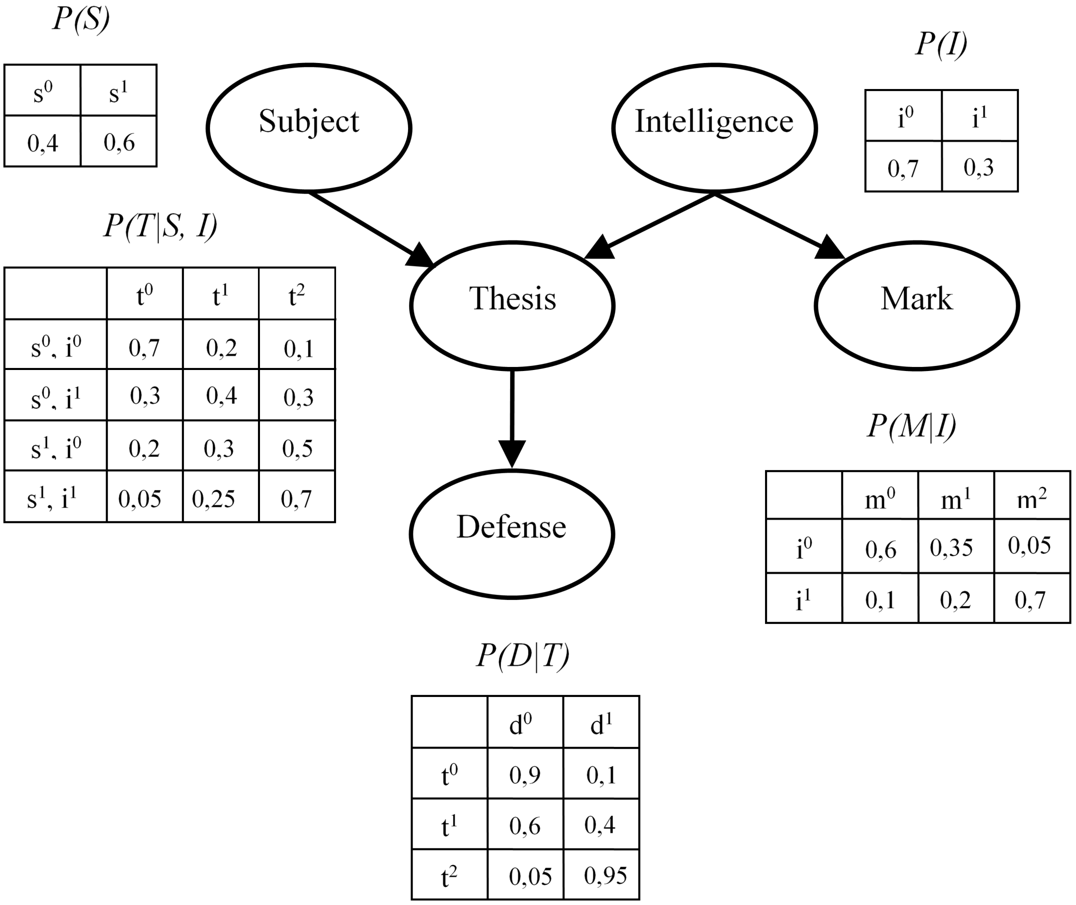
This chapter introduces the main concepts from Bayesian network framework. Basic notions are presented using a simple Bayesian network with discrete nodes in the section 3.1, the most common reasoning patterns are explained on the same network in the section 3.2. All considerations are also valid for Bayesian networks with continuous nodes, if we use appropriate type of conditional probability distribution (CPD) instead of conditional probability tables (CPTs) and specify interconnections between discrete and continuous nodes, refer to the sections 3.3 and 3.4 for more details. Dynamic Bayesian networks are introduced in the section 3.5. The introduction to inference and learning is provided in the section 3.6.

The chapter is written in introductory manner. For more information, a reader may refer to any of numerous books on Bayesian networks, e.g. [1, 87, 91, 102, 224]. Probably, the best introductory book is the one authored by F. V. Jensen and T. D. Nielsen [224] and the most comprehensive to date is the book authored by D. Koller and N. Friedman [1].

3.1 Representation and Basic Notions

A Bayesian network is a probabilistic graphical model represented by a directed acyclic graph. This model provides an opportunity to describe probability distribution over enormous number of variables in a compact way. That's what the word "probabilistic" stands for. The word "graphical" reflects the fact that a model is represented by a graph (that can be also viewed as a network). The word "model" indicates our attempt to describe a real-world process by a mathematical approximation. A probabilistic graphical model consists of the nodes that represent random variables and edges that reflect interconnections between those random variables. A fact that a graph is "directed" tells us that its edges have defined direction. However, to avoid misunderstanding, it is important to emphasize, that directed edges do not imply one-way direction of a probabilistic flow between variables. They reflect the causality of random variables (this concept will be explained in more details later in this chapter). And finally, the word "acyclic" means, that an edge in BN cannot have the same initial and target node. Moreover, all possible paths in a network has to connect two different nodes (no cycles are allowed).

As it was already mentioned, BN describes multivariate probability distribution in a compact way. But how do we get this compactness? It is achieved by using conditional probability distribution for each variable instead of trying to construct the joint probability distribution over all variables in a network. The best way to understand this concept (and further basic notions in Bayesian network framework) is to use an example.



P(S)

s^0	s^1
0,4	0,6

P(I)

i^0	i^1
0,7	0,3

P(T|S, I)

	t^0	t^1	t^2
s^0, i^0	0,7	0,2	0,1
s^0, i^1	0,3	0,4	0,3
s^1, i^0	0,2	0,3	0,5
s^1, i^1	0,05	0,25	0,7

P(M|I)

	m^0	m^1	m^2
i^0	0,6	0,35	0,05
i^1	0,1	0,2	0,7

P(D|T)

	d^0	d^1
t^0	0,9	0,1
t^1	0,6	0,4
t^2	0,05	0,95

Figure 3.1: Simple example of a Bayesian network

Consider the following issue (this is a remastered example from [1]): I would like to reason about my chances to succeed on the defense of my thesis. For simplicity, let us assume that all variables of our interest are discrete. The variable Defense describes the result of my defense. This variable is naturally influenced by the quality of my thesis represented by a variable Thesis. Now let us consider variables that may affect the quality of my thesis. Of course, we can think about a lot of factors that may have influence, but to keep things simple, let us assume that the quality of my thesis depends on two variables (from some point of view they are decisive): on the quality of research subject (variable Subject) and on my intelligence (variable Intelligence). To make things even more interesting, let us add one extra variable Mark to our network. This variable corresponds to the average mark from the master's study. Of course, average mark is a continuous variable, but we can consider three intervals of average marks that will transform a continuous variable into a discrete one. For simplicity, let us assume that variables of our interest can have 2 or 3 possible states. Information about all variables that we will reason about can be found in the table 3.1.

Table 3.1: Variables in example of a Bayesian network

Variable	**Label**	**Possible states**
Subject	S	$2(s^0 - bad/s^1 - good)$
Intelligence	I	$2(i^0 - low/i^1 - high)$
Thesis	T	$3(t^0 - poor/t^1 - good/t^2 - excellent)$
Defense	D	$2(d^0 - fail/d^1 - success)$
Mark	M	$3(m^0 - low/m^1 - medium/m^2 - high)$

Considering the amount of states for each variable, we can easily calculate that the joint probability distribution will have 72 rows (it is equal to the amount of possible combinations of variable states). Therefore, even for such small quantity of variables we would need a long table to describe the joint probability distribution. Instead, we will use Bayesian network framework. The BN that corresponds to our example is on the figure 3.1. Ovals signify nodes (random variables), arrows signify edges (direction of causal influence). Nodes can be discrete or continuous, depending on the type of random variable that they represent. Also, nodes can be observed or non-observed. In the former case, its conditional probability table is replaced by its value, which is certain (its probability is equal to one), in the latter case, its value is represented by the corresponding probability distribution. If the node cannot be observed in a network, it is called hidden.

Now let us look closer at the edges of the acyclic graph on the figure 3.1. Since the graph is directed, edges are represented by arrows. In contradiction with neural networks, these arrows do not represent the flow of information, but causal connections between random variables [225].

The initial node at each edge is said to be a parent of the target node of the same edge. For example, the node Subject is a parent of the node Thesis. Logically, node Thesis is said to be a child of the node Subject. Children are dependent on their parents in real life and so they are in a Bayesian network. Analogically, we can state, that the node Defense is a child of the node Thesis, and the node Mark is a child of the node Intelligence. The node Thesis has two parents: the node Subject and the node Intelligence.

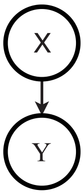
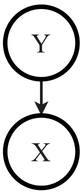
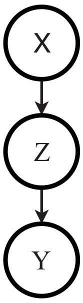
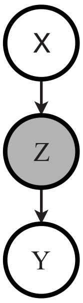
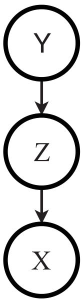
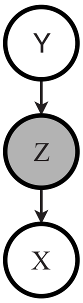
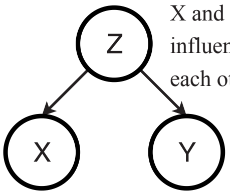
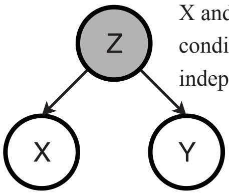
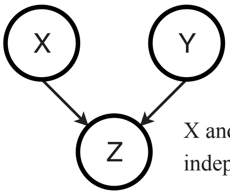
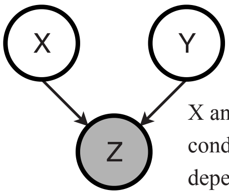
There are two broader notions that describe probabilistic dependencies between nodes in a Bayesian network: an ancestor and a descendant. The ancestor is the parent of a node, the parent of a parent and so on; the descendant is the child of a node, the child of a child and so on. In our network, nodes Thesis and Defense are descendants of the node Subject. The node Intelligence is an ancestor for nodes Thesis, Defense and Mark.

Another very important information regarding relations between random variables that can be read from the structure of a Bayesian network are independencies between variables. To understand this concept let us analyze fundamental types of connections between two nodes that are represented in the table 3.2. More complicated structures can be than divided into these basic structures and treated separately.

The first type of connection is the parent-child connection. In our network on the figure 3.1 this pair is represented, for example by the pair Subject-Thesis. The better the subject, the higher the probability that the thesis will be good. This connection is obvious. It properly works also in the other direction, from a child to its parent. The better the thesis, the higher our beliefs that the subject was good.

The third and the fourth type of connection are a little bit more complicated, but they are still quite straightforward. However, some interesting change happens after observing a variable inside the chain of variables. Let us look on the chain Subject-Thesis-Defense. Does the subject influence the probability to succeed during the defense? Surely it does. The better the subject, the higher the probability of the better thesis and consequently, the higher my chances to succeed during the defense (in fact, we used the parent-child connection twice). Let us check out, how does it work in the opposite direction, in the chain Defense-Thesis-Subject. If I succeed during the defense, your beliefs that my thesis is good will grow, so as the beliefs that I had a good topic (in this case, we used the child-parent connection twice). Now let us make things more interesting and let us assume that the variable Thesis is observed (see the second column of the table 3.2 for the third and the fourth types of connections). If I know the state of the variable Thesis, then there will be no connection between variables Subject and Defense anymore, in any direction, these

Table 3.2: Independencies in Bayesian networks

1) X is a parent of Y	**2) X is a child of Y**
 X and Y influence each other	 Y and X influence each other
3) X is an ancestor of Y	
Z is not observed	Z is observed
 X and Y influence each other	 X and Y are conditionally independent
4) X is a descendant of Y	
Z is not observed	Z is observed
 X and Y influence each other	 X and Y are conditionally independent
5) X and Y are children of the same parent	
Z is not observed	Z is observed
 X and Y influence each other	 X and Y are conditionally independent
6) X and Y are parents of the same child (V -structure)	
Z is not observed (neither is any of its descendants)	Z is observed (or any of its descendants)
 X and Y are independent	 X and Y are conditionally dependent

variables become conditionally independent. How did that happen? If you know that my thesis is bad, you will assume that probably I will fail during defense. And even if I try to convince you, that my research topic is good, this fact will not make you change your mind. It works analogously in the opposite direction. This time let us consider more optimistic scenario. If you find my thesis good, your beliefs that the subject is good will increase. And information about the result of defense will not influence these beliefs.

The fifth type of connection shown in the table 3.2 is the child-parent-child connection. In this case we use the child-parent dependence and then the parent-child dependence. Let us consider the chain Thesis-Intelligence-Mark. The worse my thesis, the worse your thoughts about my intelligence. The worse your thoughts about my intelligence, the less your expectation regarding my average mark. But if someone tells you that my average mark during master's study was high, your thoughts about my intelligence will be better and consequently, the probability of having a good thesis will increase.

The last type of connection shown in the table 3.2 is a so-called V-structure, the parent-child-parent connection. Let us consider the chain Subject-Thesis-Intelligence. The quality of a research subject is independent on my intelligence. But if we observe the middle variable, things will change. If you know that my thesis is bad and that I am smart, your beliefs that the subject is bad will increase. In the opposite direction, if you know that my thesis is bad, and you know that the research subject was good, then your beliefs in my intelligence will sadly go down.

The joint probability distribution can be found using the chain rule. Usually we would do it as follows:

$$P(S, I, T, D, M) = P(S) \cdot P(I|S) \cdot P(T|S, I) \cdot P(D|S, I, T) \cdot P(M|S, I, T, D). \quad (3.1)$$

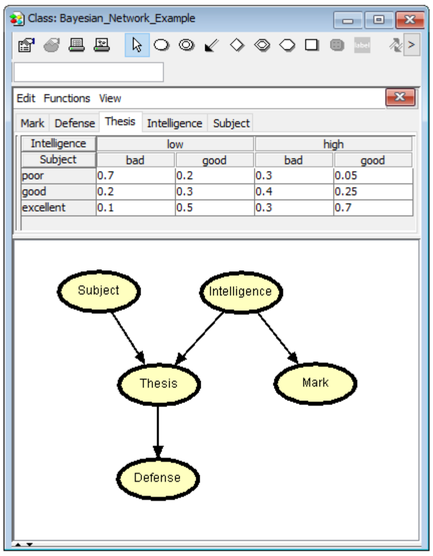
The chain rule for a Bayesian network is derived from a usual chain rule (3.1) by applying all independencies (conditional and non-conditional) that follows from the structure of a network. $P(I|S) = P(I)$, because S and I are independent. $P(D|S, I, T) = P(D|T)$, because D is conditionally independent on both S and I when the variable T is observed. $P(M|S, I, T, D) = P(M|I)$, because M is conditionally independent on all other variables in the network, when the variable I is observed. The chain rule for Bayesian network can be consequently rewritten as a product of all conditional probability distributions in the network:

$$P(S, I, T, D, M) = P(S) \cdot P(I) \cdot P(T|S, I) \cdot P(D|T) \cdot P(M|I). \quad (3.2)$$

3.2 Reasoning in Bayesian Networks

There are the plenty of software packages that are used to create Bayesian networks and to provide inference and/or learning, refer for example to the list provided by K. Murphy [182] for the most comprehensive overview available online. Some of these packages are commercial, others are open-source. Each software has advantages and disadvantages that specify the type of networks or/and tasks it can be used for.

Examples in this chapter are provided in the Hugin Lite software, free version of the Hugin software [226]. Hugin software is a user-friendly package with nice displaying properties that can be easily used for reasoning in relatively simple Bayesian networks. However, due to restrictions in the available types of variables and functions, it cannot be used to solve sophisticated issues (e.g. system identification). On the figure 3.2 you can find a network for the defense example introduced earlier in this chapter. This network is used to portray main reasoning patterns used in Bayesian networks.



Class: Bayesian_Network_Example

Intelligence	low		high	
Subject	bad	good	bad	good
poor	0.7	0.2	0.3	0.05
good	0.2	0.3	0.4	0.25
excellent	0.1	0.5	0.3	0.7

Figure 3.2: Example of a Bayesian network in Hugin software

Very important question is how to consider probabilities of random variables in Bayesian networks. Let us provide some principal calculations in our example network from the figure 3.1 and compare them with the results from simulation.

There are four essential reasoning patterns that can be applied in a Bayesian network:

1. **Prior reasoning**. This pattern allows us to reason about random variables without any observed evidence.

2. **Causal reasoning**. This pattern allows us to assess probabilities of possible outcomes (provides predictions).
3. **Evidential reasoning**. This pattern allows us to search for a reason of certain outcomes (provides diagnosis).
4. **Intercausal reasoning**. This pattern allows us to update our beliefs about one of possible reasons of an outcome given information about other possible reason and outcome that they share.

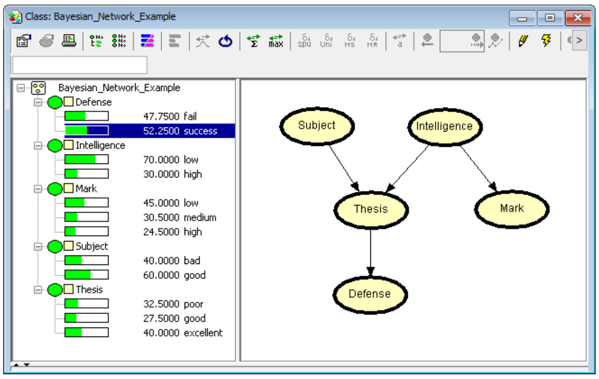
In the further explanation let us look closer on each pattern.

Prior reasoning

Prior reasoning is applicable when we want to find the probability of the certain state of a random variable (or the combination of states) without any evidence that might change our prior beliefs. For example, I would like to know, what my prior chances to defend my thesis are. To answer this question, we have to calculate marginal probability $P(d^1)$. Using the chain rule for our Bayesian network (3.2), we get the following probability:

$$P(d^1) = \sum_{S,I,T,M} P(S) \cdot P(I) \cdot P(T|I,S) \cdot P(M|I) \cdot P(d^1|T) = 0.5225. \tag{3.3}$$

Another way to find this value is to look on the prior probability of successful defense in the network, see figure 3.3.



```
Bayesian_Network_Example
  Defense
    47.7500 fail
    52.2500 success
  Intelligence
    70.0000 low
    30.0000 high
  Mark
    45.0000 low
    30.5000 medium
    24.5000 high
  Subject
    40.0000 bad
    60.0000 good
  Thesis
    32.5000 poor
    27.5000 good
    40.0000 excellent
```

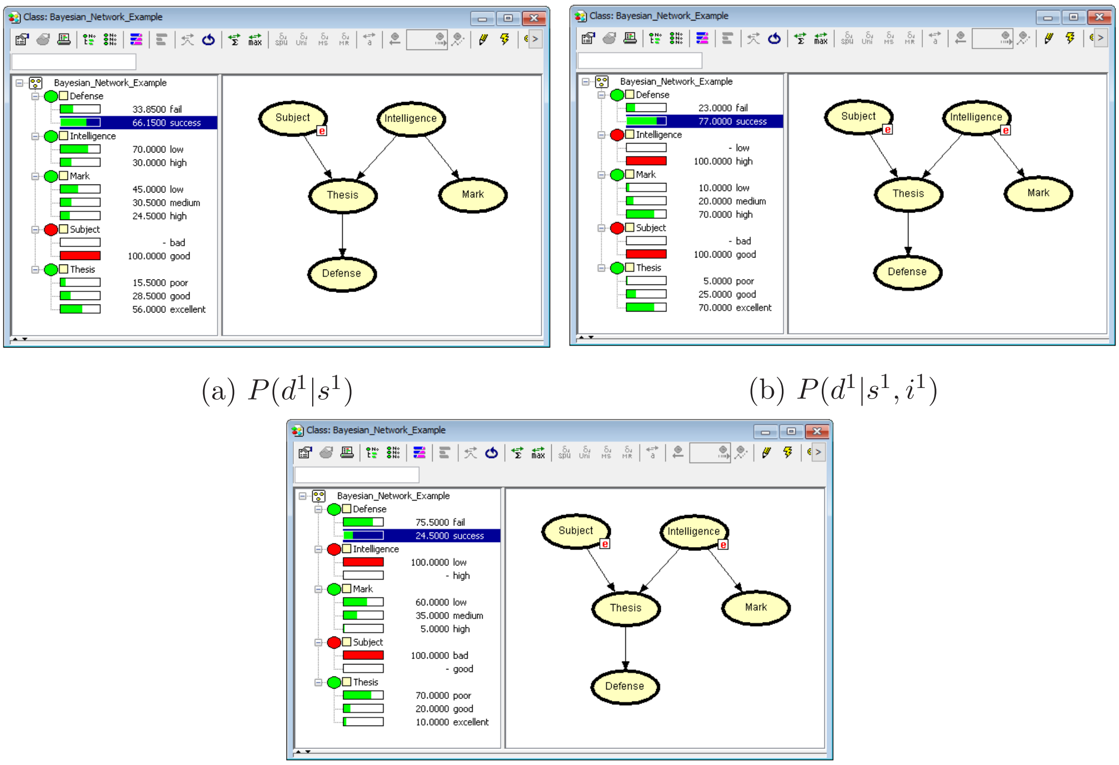
Figure 3.3: Prior reasoning in the example Bayesian network

Causal reasoning

In this pattern, we are interested in the consequences of an event. We have already found out, that my chances to defend the thesis successfully are approximately 52%. How will our beliefs change, if we discover that a subject is good? To find this, we can calculate the following conditional probability:

$$P(d^1|s^1) = \frac{P(d^1, s^1)}{P(s^1)} = \frac{\sum_{I,T,M} P(s^1) \cdot P(I) \cdot P(T|I, s^1) \cdot P(M|I) \cdot P(d^1|T)}{P(s^1)} = 0.6615. \tag{3.4}$$

We can see that my chances increased. We could also find this value from the network by entering evidence $S = s^1$ and discovering a posterior marginal probability of successful defense, see figure 3.4a.



```
Class: Bayesian_Network_Example
Bayesian_Network_Example
  Defense
    33.8500 fail
    66.1500 success
  Intelligence
    70.0000 low
    30.0000 high
  Mark
    45.0000 low
    30.5000 medium
    24.5000 high
  Subject
          - bad
    100.0000 good
  Thesis
    15.5000 poor
    28.5000 good
    56.0000 excellent
```

(a) $P(d^1|s^1)$

```
Class: Bayesian_Network_Example
Bayesian_Network_Example
  Defense
    23.0000 fail
    77.0000 success
  Intelligence
          - low
    100.0000 high
  Mark
    10.0000 low
    20.0000 medium
    70.0000 high
  Subject
          - bad
    100.0000 good
  Thesis
    5.0000 poor
    25.0000 good
    70.0000 excellent
```

(b) $P(d^1|s^1, i^1)$

```
Class: Bayesian_Network_Example
Bayesian_Network_Example
  Defense
    75.5000 fail
    24.5000 success
  Intelligence
    100.0000 low
          - high
  Mark
    60.0000 low
    35.0000 medium
    5.0000 high
  Subject
    100.0000 bad
          - good
  Thesis
    70.0000 poor
    20.0000 good
    10.0000 excellent
```

(c) $P(d^1|s^0, i^0)$

Figure 3.4: Causal reasoning in the example Bayesian network

Let us now add one more evidence to the network. Consider the most optimistic scenario of having a good subject and simultaneously of having a high intelligence. In this case we are interested in the calculation of the following conditional probability:

$$\begin{gathered}P(d^1|s^1,i^1)=\frac{P(d^1,s^1,i^1)}{P(s^1,i^1)}=\\ \frac{\sum_{T,M}P(s^1)\cdot P(i^1)\cdot P(T|i^1,s^1)\cdot P(M|i^1)\cdot P(d^1|T)}{P(s^1,i^1)}=0.77.\end{gathered}\tag{3.5}$$

My chances again increased. We can find this probability in the network by adding one more evidence $I=i^1$, see figure 3.4b.

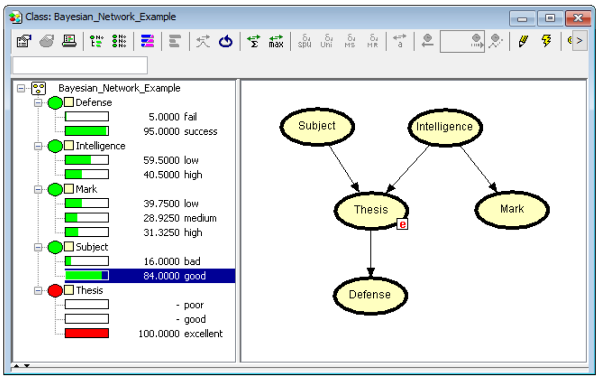
For comparison, let us consider the most pessimistic scenario of my defense. If my research topic is bad and I am not intelligent, then my chances of successful defense will considerably decrease. To prove this fact, let us calculate corresponding conditional probability (results from simulation are on the figure 3.4c).

$$P(d^1|s^0,i^0)=\frac{P(d^1,s^0,i^0)}{P(s^0,i^0)}=\sum_T P(T|i^0,s^0)\cdot P(d^1|T)=0.245.\tag{3.6}$$

Evidential reasoning

In this pattern, we are interested in reasons of a certain event. Consider the following example. You concluded that a thesis is excellent, and you are interested what are the chances that the research topic was good. To answer this question, we should calculate the following conditional probability:

$$\begin{gathered}P(s^1|t^2)=\frac{P(s^1,t^2)}{P(t^2)}=\\ \frac{\sum_{I,D,M}P(s^1)\cdot P(I)\cdot P(t^2|s^1,I)\cdot P(D|t^2)\cdot P(M|I)}{\sum_{S,I,D,M}P(S)\cdot P(I)\cdot P(t^2|S,I)\cdot P(D|t^2)\cdot P(M|I)}=0.84.\end{gathered}\tag{3.7}$$



```
Bayesian_Network_Example
  Defense
          5.0000 fail
         95.0000 success
  Intelligence
         59.5000 low
         40.5000 high
  Mark
         39.7500 low
         28.9250 medium
         31.3250 high
  Subject
         16.0000 bad
         84.0000 good
  Thesis
               - poor
               - good
        100.0000 excellent
```

Figure 3.5: Evidential reasoning in the example Bayesian network

The probability of a good research topic given the evidence that the topic was good is 84% (the prior probability for this event was 60%). Now let us find out what are my chances to have low intelligence, if my thesis is excellent:

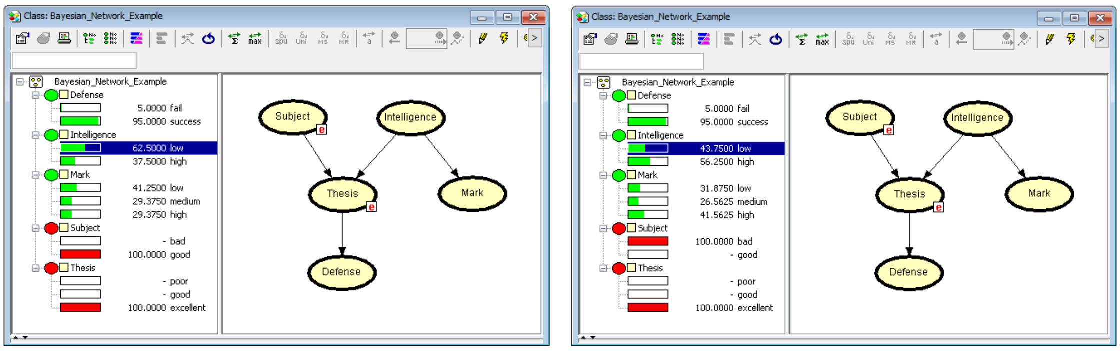
$$
\begin{gathered}
P(i^0|t^2) = \frac{P(i^0, t^2)}{P(t^2)} = \\
\frac{\sum_{S,D,M} P(S) \cdot P(i^0) \cdot P(t^2|S, i^0) \cdot P(D|t^2) \cdot P(M|i^0)}{\sum_{S,I,D,M} P(S) \cdot P(I) \cdot P(t^2|S, I) \cdot P(D|t^2) \cdot P(M|I)} = 0.595.
\end{gathered}
\tag{3.8}
$$

My chances to be stupid will be 59.5%. Quite high probability, but it is lower than the prior one (70%), so the network works properly. Both probabilities can be found in the network as posterior marginal probabilities given the evidence $T = t^2$, see figure 3.5.

Intercausal reasoning

This pattern is not as intuitive as other ones, but it is particularly interesting and useful. Let us see, how two conditionally dependent variables, the Intelligence and the Subject, influence each other when the variable Thesis is observed. We have already seen that my chances to be stupid if my thesis is excellent are 59.5%. But how will these beliefs change, if you discover that the subject was good? To answer this question, let us calculate the following conditional probability:

$$
\begin{gathered}
P(i^0|t^2, s^1) = \frac{P(i^0, t^2, s^1)}{P(t^2, s^1)} = \\
\frac{\sum_{D,M} P(s^1) \cdot P(i^0) \cdot P(t^2|s^1, i^0) \cdot P(D|t^2) \cdot P(M|i^0)}{\sum_{I,D,M} P(s^1) \cdot P(I) \cdot P(t^2|s^1, I) \cdot P(D|t^2) \cdot P(M|I)} = 0.625.
\end{gathered}
\tag{3.9}
$$



```
Class: Bayesian_Network_Example
Bayesian_Network_Example
  Defense
      5.0000 fail
     95.0000 success
  Intelligence
     62.5000 low
     37.5000 high
  Mark
     41.2500 low
     29.3750 medium
     29.3750 high
  Subject
           - bad
    100.0000 good
  Thesis
           - poor
           - good
    100.0000 excellent
```

(a) $P(i^0|t^2, s^1)$

```
Class: Bayesian_Network_Example
Bayesian_Network_Example
  Defense
      5.0000 fail
     95.0000 success
  Intelligence
     43.7500 low
     56.2500 high
  Mark
     31.8750 low
     26.5625 medium
     41.5625 high
  Subject
    100.0000 bad
           - good
  Thesis
           - poor
           - good
    100.0000 excellent
```

(b) $P(i^0|t^2, s^0)$

Figure 3.6: Intercausal reasoning in the example Bayesian network

My chances increased, because the variable Subject has already given you a reliable explanation why the thesis is excellent, that's why the second possible reason became less probable, see figure 3.6a.

And finally, let us see how this probability will change with the converse observation of the variable Subject:

$$P(i^0|t^2, s^0) = \frac{P(i^0, t^2, s^0)}{P(t^2, s^0)} = \frac{P(i^0) \cdot P(t^2|s^0, i^0)}{\sum_I P(I) \cdot P(t^2|s^0, I)} = 0.4375. \tag{3.10}$$

See figure 3.6b for comparison with the result of simulation.

3.3 Bayesian networks with continuous nodes

In the previous sections we dealt with a Bayesian network that contained only discrete nodes described by conditional probability tables. But we often need to add a continuous node to the structure of a network. Moreover, in some cases we want to reason about the continuous variables only. Therefore, it is important to formulate the description of continuous nodes in a Bayesian network and combinations of continuous and discrete nodes.

A continuous variable can be represented by probability density function that depends on the probability distribution of a random variable. In this thesis, we assume that all continuous variables follow the normal distribution. This assumption is often used for Bayesian networks with continuous nodes. It is important to point out, that this assumption does not ruin the generality of proposed approaches. For variables that are not normally distributed, the mixture of normal distributions can be used as the approximation of an arbitrary continuous distribution, see section 3.4 for details.

The probability density of the Gaussian distribution is:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp -\frac{(x-\mu)^2}{2\sigma^2}. \tag{3.11}$$

In the equation (3.11), μ is the mean or expectation of the distribution and σ^2 is the variance.

The shape of Gaussian distribution is fully defined by its mean and variance. Therefore, for normally distributed variables, it is common to represent probability distribution using the following notion:

$$x \sim \mathcal{N}(\mu, \sigma^2). \tag{3.12}$$

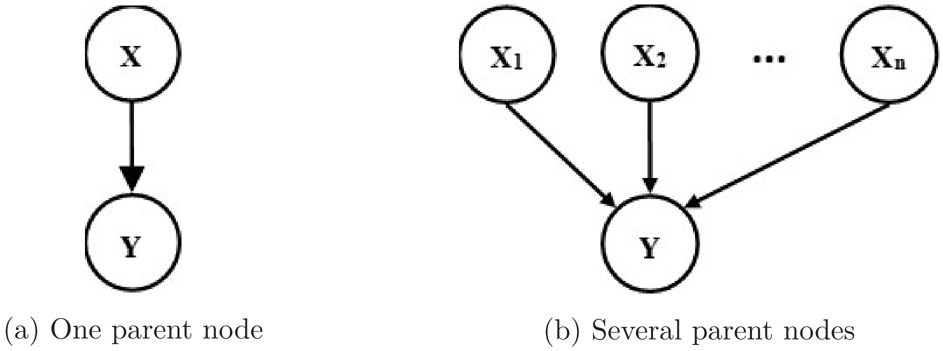
If the variable of our interest is not a scalar, but a vector variable, then its behavior can be described by multinomial normal distribution. In this case we consider several variables that are jointly normal, the description of probability distribution requires knowing of the means of all variables and a covariance matrix. Multinomial normal distribution can be represented using the following notion:

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} \mu_{x_1} & \mu_{x_2} & \cdots \mu_{x_n} \end{bmatrix}, \Sigma_X). \tag{3.13}$$

In the equation (3.13), x_i are considered random variables, μ_{x_i} are corresponding means and Σ_X is a covariance matrix that consists of variances of each random variable and covariances between each pair of variables:

$$\Sigma_X = \begin{pmatrix} \sigma^2_{x_1} & \sigma_{x_1 x_2} & \cdots & \sigma_{x_1 x_n} \\ \sigma_{x_1 x_2} & \sigma^2_{x_2} & \cdots & \sigma_{x_2 x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{x_1 x_n} & \sigma_{x_2 x_n} & \cdots & \sigma^2_{x_n} \end{pmatrix}. \tag{3.14}$$

It is also important to define the types of distributions for different type of parent nodes for a continuous child node (i.e. continuous or discrete), see figure 3.7.



(a) One parent node

(b) Several parent nodes

Figure 3.7: Continuous node with one or more parent nodes

Bayesian network framework does not put any restrictions neither on the type of used probability distribution, nor on the type of interconnections between nodes. The conditional probability distributions described in this section correspond to the most frequently used types of distributions that are currently supported by software packages for Bayesian networks.

Discrete node → continuous node

If a continuous node is a child of a single discrete node (see figure 3.7a), then depending on the state of a parent node, the parameters of its probability distribution vary. For the simplest case of binary discrete node, a continuous node has the following distribution:

$$\begin{cases} Y|X \sim \mathcal{N}(\mu_1, \sigma_1^2) \text{ if } X = 0; \\ Y|X \sim \mathcal{N}(\mu_2, \sigma_2^2) \text{ if } X = 1. \end{cases} \tag{3.15}$$

In general, the amount of possible probability distributions that we have to define for a continuous node is equal to the amount of possible states of a parent discrete node.

If a continuous node has several discrete parents (see figure 3.7b), then it will have the following probability distribution (assuming that all discrete variables are binary):

$$\begin{cases} Y|X_1, X_2, \cdots, X_n \sim \mathcal{N}(\mu_1, \sigma_1^2) \text{ if } X_1 = 0, X_2 = 0, \cdots, X_n = 0; \\ Y|X_1, X_2, \cdots, X_n \sim \mathcal{N}(\mu_2, \sigma_2^2) \text{ if } X_1 = 0, X_2 = 0, \cdots, X_n = 1; \\ \cdots \\ Y|X_1, X_2, \cdots, X_n \sim \mathcal{N}(\mu_N, \sigma_N^2) \text{ if } X_1 = 1, X_2 = 1, \cdots, X_n = 1. \end{cases} \tag{3.16}$$

In (3.16) n is the amount of discrete parent nodes and N is the amount of possible probability distributions that we have to define for a continuous node. This value is equal to the total amount of combinations of possible states of parent discrete nodes.

Continuous node → continuous node

If a continuous node is a child of a single continuous node (see figure 3.7a), then we have to specify the type of function that describes their dependence. The most common choice is linear dependence. This choice is also supported in software packages for Bayesian networks. The probability distribution of a child node with a continuous parent can be written as:

$$Y|X \sim \mathcal{N}(\lambda_0 + \lambda_1 \cdot X, \sigma_y^2). \tag{3.17}$$

If a continuous node has several continuous parents (see figure 3.7b), then its distribution is represented by the following conditional probability distribution:

$$Y|X_1, X_2, \cdots, X_n \sim \mathcal{N}(\lambda_0 + \lambda_1 \cdot X_1 + \lambda_2 \cdot X_2 + \cdots + \lambda_n \cdot X_n, \sigma_y^2). \tag{3.18}$$

If a continuous child node has both continuous and discrete parents, then the conditional probability distribution is a combination of (3.16) and (3.18). The amount of possible distributions is defined by the amount of possible combinations of discrete parent nodes, and for each of these combinations, the weighted sum of the values of continuous parent nodes (3.18) with different weights $\beta_0, \beta_1, \cdots, \beta_n$ and different value of variance is specified.

If a considered network has discrete nodes with continuous parents, then the appropriate type of conditional probability distribution for this type of relations has to be also specified. The common choices are softmax function (multinomial logit function) and multi-layer perceptron (neural network node).

3.4 Nodes corresponding to the mixture of normal distributions

As it has been already mentioned above, for continuous variables, which are not distributed normally, the mixture of normal distributions can be used as the approximation of almost any continuous distribution. The basic idea is to divide a target distribution into parts that can be assumed to be normally distributed and combine them into one distribution. Fusion of several normally distributed components into one probability distribution is provided by the computation of weighted average with weights $\omega_1, \omega_2, \cdots, \omega_n$ that fulfill the following conditions:

$$\omega_i \in (0,1); \sum_{i=1}^{n} \omega_i = 1. \tag{3.19}$$

Weights identify, how strong influence each chosen normal component distribution has on the target distribution.

Assume that we divided a target distribution into the following distributions:

$$\begin{gathered} x_1 \sim \mathcal{N}(\mu_{x_1}, \sigma_{x_1}^2), \\ x_2 \sim \mathcal{N}(\mu_{x_2}, \sigma_{x_2}^2), \\ \cdots \\ x_n \sim \mathcal{N}(\mu_{x_n}, \sigma_{x_n}^2) \end{gathered} \tag{3.20}$$

with weights $\omega_1, \omega_2, ...\omega_n$. The mixture of normal distributions can be consequently defined as:

$$y \sim \mathcal{N}(\omega_1 \cdot \mu_{x_1} + \omega_2 \cdot \mu_{x_2} + \cdots + \omega_n \cdot \mu_{x_n}, \sigma_y^2). \tag{3.21}$$

The variance σ_y can be found as:

$$\sigma_y^2 = \sum_{i=1}^{n} \omega_i \cdot \sigma_i^2 + \sum_{i=1}^{n} \omega_i \cdot \mu_i^2 - (\sum_{i=1}^{n} \omega_i \cdot \mu_i)^2. \tag{3.22}$$

In the Bayesian network framework a node that corresponds to the mixture of normal distributions can be represented by a combination of a hidden discrete node and a continuous node (see figure 3.8).

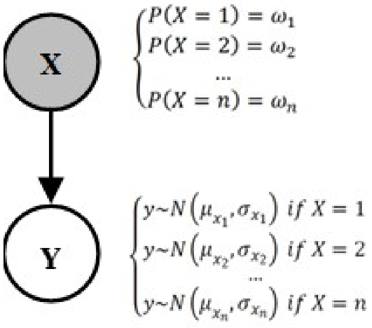

X
Y
$P(X=1)=\omega_1$
$P(X=2)=\omega_2$
...
$P(X=n)=\omega_n$
$y \sim N(\mu_{x_1}, \sigma_{x_1})$ if $X=1$
$y \sim N(\mu_{x_2}, \sigma_{x_2})$ if $X=2$
...
$y \sim N(\mu_{x_n}, \sigma_{x_n})$ if $X=n$

Figure 3.8: A Gaussian mixture node

The amount of possible states for a discrete node is given by the amount of components in the mixture of normal distributions. The probabilities of each state are considered to be equal to weights. Probability distributions that are defined in a continuous node for each state are equal to initial distributions of components that correspond to appropriate weight in the mixture model.

Since the discrete node is hidden, it can never be observed. Consequently, the probability distribution of a continuous node is represented by a weighted average of initial normal distributions.

3.5 Dynamic Bayesian Networks

A Bayesian network that was used as an example in the sections 3.1 and 3.2 is considered to be a static network. The word “static” used in this context means that the probability distribution of any random variable depends only on variables at the same moment in time. But sometimes we are interested not only in the values of variables at current time step, but also in the previous one, or even in several previous time steps, i.e. temporal dependencies between random variables. In this case we have to use the dynamic (temporal) extension of Bayesian networks.

A dynamic Bayesian network (DBN) is a probabilistic graph, in which we contemplate “copies” of the same variables in different time steps, interconnections

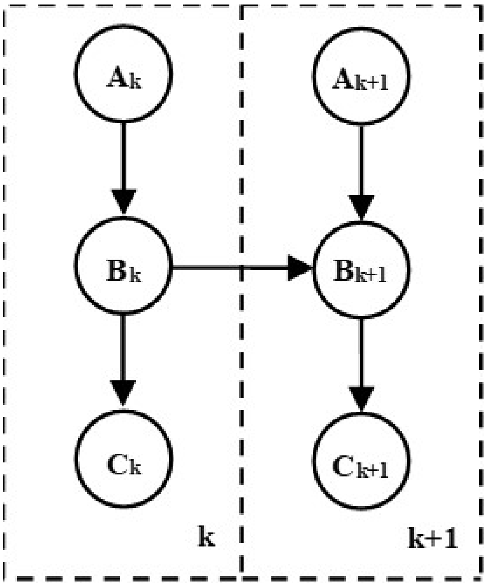


Figure 3.9: Example of a dynamic Bayesian network

between these variables are described by a unified conditional probability distributions for each "copy" of a variable. If variables in a network depend only on variables at current time step and the previous one, then this network is said to be a two-time-slices DBN (2TDBN). In general, if the current state in a network depends on variables at n previous steps, then this network is a (n+1)TDBN.

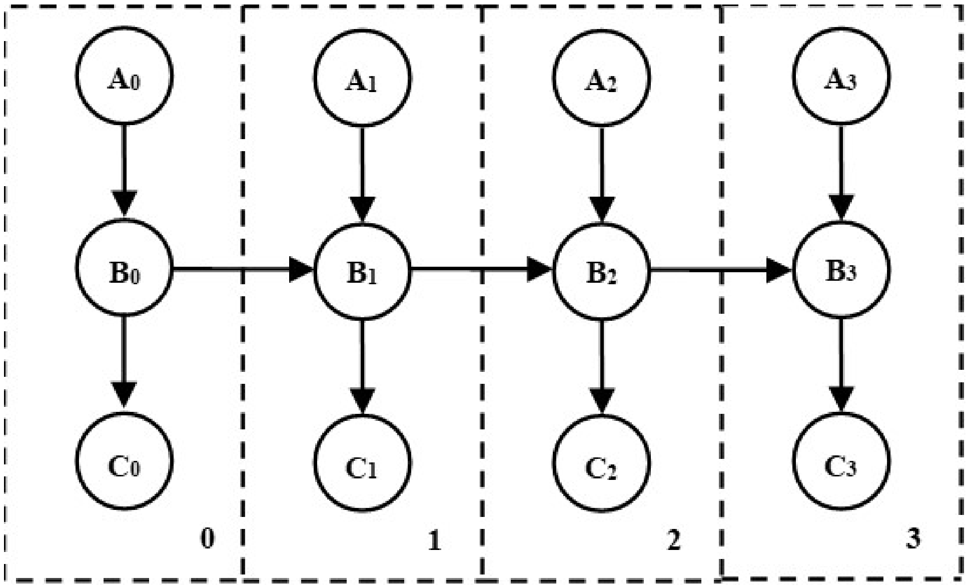


Figure 3.10: Example of an unrolled dynamic Bayesian network

An example of 2TDBN is shown on the figure 3.9. This network has 3 random variables, A, B and C. Variable A at each time step does not depend on any other variable, variable C depends only on the value of variable B at the same time step. The variable B depends on the variable A and also on the previous value of variable B (except of the first time slice, when it depends only on the variable A). For better understanding of mutual interconnections between random variables in following steps it may be useful to draw so-called unrolled (or grounded) Bayesian network that is a static representation of a DBN. The unrolled BN for the network on the figure 3.9 for first four time slices is shown on the figure 3.10.

A 2TDBN can be considered a Markovian system that satisfies Markov independence assumption [1]:

$$X^{(t+1)} \perp X^{(0:(t-1))} | X^{(t)}. \tag{3.23}$$

This assumption reflects an independence property that can be also formulated verbally as follows: values of variables at the time step (t+1) are independent of all previous values given the state at the time t.

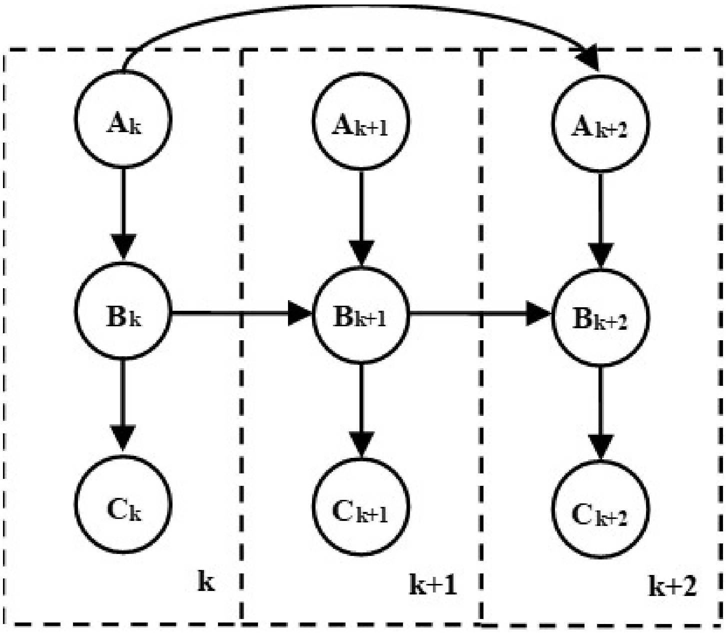


Figure 3.11: Example of a 3TDBN

Representation of a 2TDBN requires knowing of the initial values of variables and interconnections between them and the description of a transition model between two adjacent time steps.

The transition model of a DBN can also represent interconnections between more than two adjacent time steps. In this BN, values of one or more variables depend not only on the values in the previous step, but also on “older” values. An example of a 3TDBN is shown on the figure 3.11. In this network, variable A depends on its values in two previous steps. Definition of this network requires knowing of variables in the first and the second time slice. In general, the representation of (n+1)TDBN contains the definition of variables and interconnections between them in n first steps and the definition of interconnections between particular steps. Even though these networks are not Markovian, there are independence assumptions that simplify inference in these networks. We can formulate them as follows: the values of variables at the time step $(t+n)$ are independent on all previous values at time steps up to time step $(t-1)$ given the state at the time t:

$$X^{(t+n)} \perp X^{(0:(t-1))}|X^{(t)}. \tag{3.24}$$

3.6 Inference and learning in Bayesian networks

The calculation of conditional and marginal probabilities by the means of traditional probability theory is time-consuming even for small Bayesian network, see examples in the section 3.2. For Bayesian networks used in practice this type of calculations is often intractable. Therefore, many algorithms that provide these calculations for different types of Bayesian networks were designed. These algorithms are also called inference algorithms and the task itself is often referred to as inference.

By learning in Bayesian network framework we understand solving of one of two tasks: estimation of parameters of a Bayesian network from a known structure and evidence on nodes (parameter learning) and choosing of the best structure of a Bayesian network (structure learning).

Due to the modularity of Bayesian network framework, the techniques for constructing a Bayesian network, inference and learning are separated and not task-specific. Therefore, we can use different inference and learning algorithms for the same Bayesian network. This is very useful, since new algorithms for Bayesian networks can be easily adopted for different practical applications.

Inference

Inference algorithms for Bayesian networks can be exact or approximate. Exact inference algorithm can be used in cases, in which the solution in closed form is available. In other cases, approximate inference has to be implemented. For example, Bayesian networks that consists of continuous nodes that are normally distributed with linear dependence between parent and child nodes can be treated using exact inference algorithms. Another example is a Bayesian network, in which all hidden nodes are discrete. For complex Bayesian networks approximate inference can be more appropriate even if exact inference is possible, since it does not have to be computationally feasible.

The most popular classes of exact inference algorithms are:

- Variable elimination class. These algorithms exploit the decomposition of the joint probability distribution using the chain rule for Bayesian networks and marginalizes out unobserved nodes one by one. The complexity of inference depends on chosen elimination order.

- Message passing class. In these algorithms the nodes from Bayesian networks are divided into clusters that pass messages to each other regarding the nodes which they share in their scope. Using of the tree-structured cluster graph guarantees the correctness of obtained marginals.

The most popular classes of approximate inference algorithms are:

- Sampling (particle-based) algorithms. These algorithms sample instances from a target distribution and use them for the calculation of the quantities of our interest. The distinguished representatives of these algorithm are Markov Chain Monte Carlo methods, Gibbs sampling methods and the Metropolis-Hasting algorithm.

- Variational methods. In these methods all nodes are decoupled and a variational parameter is introduced for each of them. These parameters are updated iteratively using the procedure that minimizes the cross-entropy between the real probability distribution and its approximation.

Inference in dynamic Bayesian network can be provided by the unrolling of a network for the amount of time slices that corresponds to the length of a considered trajectory and consequent implementation of inference algorithms for static networks. The using of this approach requires preliminary fixing of parameters (parameter tying) for the same variables in different time slices. This approach can be time-consuming or even intractable for long trajectories.

Another way of inference in dynamic Bayesian networks is to use dynamic implementations of inference algorithms for static Bayesian networks.

Parameter learning

Unknown parameters in a Bayesian network can be treated as unknown constant (frequentist approach) or unknown variables (Bayesian approach). In the former case, we search for point estimate and in the latter case we search for probability distributions of unknown parameters. If we use the Bayesian approach, then the parameters are treated as random variables, similarly to other random variables in a considered network and therefore the learning task corresponds to the task of inference. If we use frequentist approach, then we search for maximum likelihood estimates (MLE) or maximum a posteriori (MAP).

If the dataset is partially observed (i.e. there are missing values in a dataset or hidden variables in the network structure), then the parameter estimation task becomes more complicated. The exact parameter posterior is in general multimodal for this setting and hence there is no guarantee to reach the global optimum. The simplest and the most frequently used algorithm for parameter estimation in partially observed Bayesian network is expectation maximization (EM) algorithm. This algorithm is iterative, each iteration has two steps: in the E step the missing values of random variables are calculated given the current guess of parameters and in the M step the parameters are calculated given the current guess of missing values using MLE or MAP. Calculation of missing values can be provided by an exact or an approximate inference algorithm. The EM algorithm stops when it reaches the local optimum. It is possible to implement other algorithms for parameter estimation in partially observed BNs, for example gradient-based methods.

Structure learning

If no information is given neither about parameters of a Bayesian network, nor about its structure, then the structure learning procedure has to be used. There are two approaches to structure learning: constrained based and search-and-score. In the former one, we start from the fully connected graph and remove edges from it if the corresponding conditional independence between random variable is reveled from the data. In the latter one, we search over the set of possible structures to find the one that describes best the joint probability distribution over the variables in the scope. The fit of the structure to the dataset is assessed using a scoring function, e.g. likelihood score or Bayesian information criterion (BIC). Searching over all possible structures is infeasible for the majority of practical applications, since the number of possible DAGs is super-exponential in the number of considered nodes. Therefore, it is important to reduce the set of possible DAGs before structure learning using expert knowledge regarding interconnection between considered random variables.

4 Modelling of dynamic systems using Bayesian networks

This chapter provides bridging between the well-known models of dynamic systems (i.e. difference equation and state space representation) and Bayesian networks. The resulting networks correspond to stochastic discrete-time description of dynamic systems. The section 4.1 links difference equation with Bayesian networks. The section 4.2 is dedicated to the linkage between state space representation and Bayesian networks. Available software packages for Bayesian networks have restrictions on the type of used nodes and structures. To overcome some of these limitations, the implementation of models with both static and dynamic networks is considered. In addition, the implementation of multivariate nodes (required for state space model structure) using the set of univariate nodes is addressed.

The models presented in this chapter can be used for simulation, monitoring and identification of dynamic systems. The DBN-based state space model of dynamic systems has been successfully used also in control [6], refer to the section 1.3 for more details. In principle, all networks presented in this chapter can be used in control of dynamic systems with implementation of feedback using the paradigm described in [6].

Bayesian network can also be used as state observers, this task is addressed in the section 4.3.

4.1 Model structures based on a difference equation

Let us rewrite the difference equation with positive shift (2.16) for the bound case $(m = n)$. The additional condition ensures that the resulting equation can still be used for linear dynamic systems with the arbitrary value of m:

$$\begin{gathered} y(k+n) + a_{n-1}y(k+n-1) + \cdots + a_1 y(k+1) + a_0 y(k) = \\ b_n u(k+n) + b_{n-1}u(k+n-1) + \cdots + b_1 u(k+1) + b_0 u(k), \\ b_i = 0 \text{ for } m < i \leq n. \end{gathered} \tag{4.1}$$

The difference equation in (4.1) can be rewritten as:

$$\begin{gathered} y(k+n) = b_n u(k+n) + b_{n-1}u(k+n-1) + \cdots + b_1 u(k+1) + b_0 u(k) \\ -a_{n-1}y(k+n-1) - \cdots - a_1 y(k+1) - a_0 y(k) = \\ -\sum_{i=1}^{n} a_i \cdot y(k+n-i) + \sum_{i=0}^{n} b_i \cdot u(k+n-i), \\ b_i = 0 \text{ for } m < i \leq n. \end{gathered} \tag{4.2}$$

The representation of dynamic systems in the domain of Bayesian networks using difference equation (4.2) is presented on the figure 4.1. It is worth to mention, that besides this transition model, we also have to specify the values of variables in the first n time slices. The resulting DBN is (n+1)TDBN, where n is the order of a considered dynamic system.

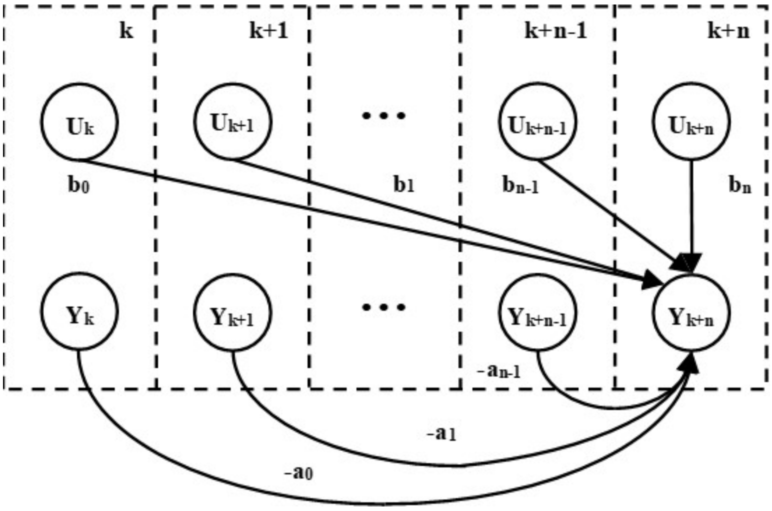


Figure 4.1: Representation of difference equation using dynamic BN

Assuming all variables are Gaussian and all dependencies are linear, the probability distributions of variables in the network from the figure 4.1 are:

$$\begin{gathered} Y_{k+n}|(Y_k, Y_{k+1}, \cdots, Y_{k+n-1}, U_k, U_{k+1}, \cdots, U_{k+n-1}) \sim \\ \mathcal{N}(\mu_Y - \sum_{i=1}^{n} a_i \cdot Y(k+n-i) + \sum_{i=0}^{n} b_i \cdot U(k+n-i), \sigma_Y^2), \\ U_k \sim \mathcal{N}(\mu_{U_k}, \sigma_U^2). \end{gathered} \tag{4.3}$$

If we set $\mu_Y = 0$, then the probability distribution of Y_{k+n} in (4.3) corresponds to the difference equation (4.1) enriched by the influence of normally distributed noise on the input with variance σ_U^2 and on the output with variance σ_Y^2.

This way of representation is quite natural for dynamic systems, because it contains all parameters in explicit form. If initial values of input and output variables are not equal to zero, they can be assigned directly to appropriate variables. On the other hand, the change of the order of a model requires significant reconstruction of a DBN (change in the amount of time slices). Moreover, the structures of networks based on a difference equation in general do not fulfill the Markov independence assumption (3.23) (except the system of the first order), since they use more than two time slices to describe system's dynamics. This type of networks has less support in software packages for Bayesian networks. For example, Bayes Net Toolbox for MATLAB used for experiments in this thesis does not support this type of DBNs.

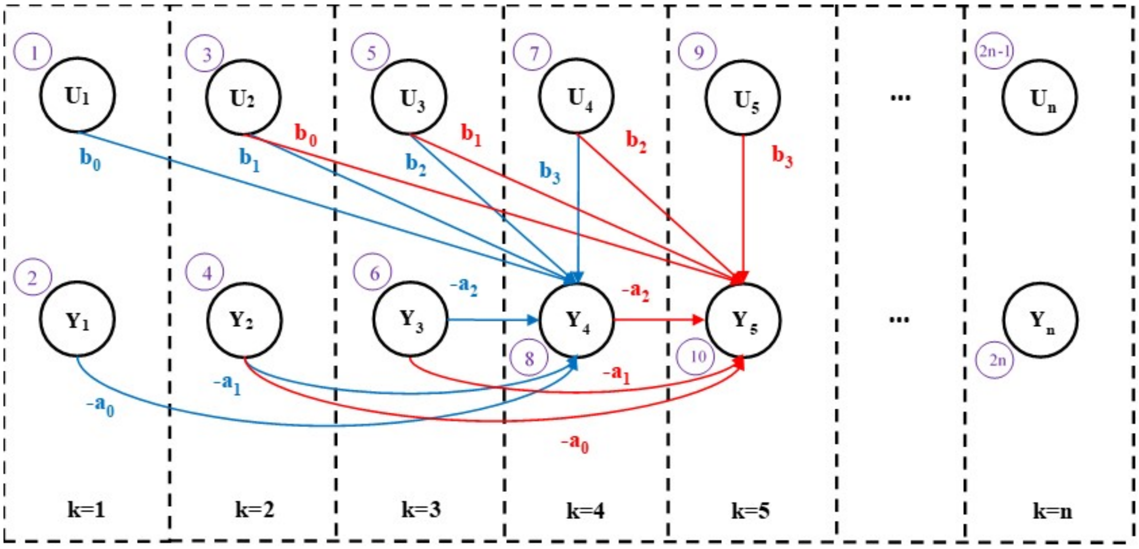


Figure 4.2: Representation of difference equation using static BN

If we want to use difference equation based model for identification, we have to measure the series of input and output variables and find parameters of a difference equation using learning algorithms for DBNs with continuous nodes.

The implementation of difference equation based structure using a static Bayesian network can be provided by the unrolling of a network presented on the figure 4.1 for T/dt time slices, where T is the length of considered signals and dt is a discretization step (sampling rate). The resulting structure will slightly differ for the different values of m and n. For example, the structure of a Bayesian network for dynamic system of the third order, where $m = n$ is presented on the figure 4.2.

Using of such model for the identification of dynamic systems is possible, if we define that all corresponding parameters in different time slices are the same (in the BNT this procedure is called parameter tying).

4.2 Model structures based on a state space representation

Representation of dynamic systems in the domain of Bayesian networks using state space representation (2.23) is presented on the figure 4.3. Assuming all variables are Gaussian and all dependencies are linear, the probability distributions of all variables in this network are:

$$\begin{aligned} \mathbb{X}_{k+1}|\mathbb{X}_k, U_k &\sim \mathcal{N}(\mu_{\mathbb{X}} + \mathbb{A}\cdot\mathbb{X}_k + \mathbb{B}\cdot U_k, \Sigma_{\mathbb{X}}), \\ Y_k|\mathbb{X}_k, U_k &\sim \mathcal{N}(\mu_Y + \mathbb{C}\cdot\mathbb{X}_k + \mathbb{D}\cdot U_k, \sigma_Y^2), \\ U_k &\sim \mathcal{N}(\mu_U, \sigma_U^2). \end{aligned} \tag{4.4}$$

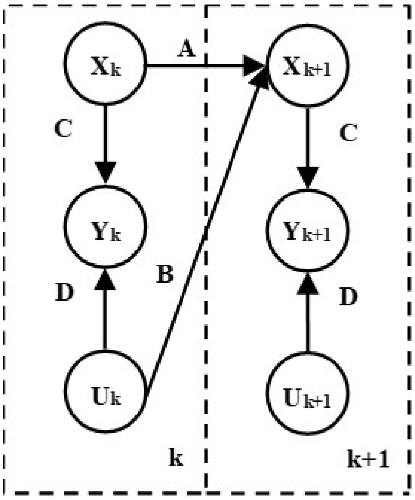


Figure 4.3: Representation of state space model using dynamic BN

If we set $\mu_Y = 0$ and $\mu_{\mathbb{X}} = 0$, then the probability distributions of Y_k and $\mathbb{X}_{k+1}$ in (4.4) correspond to the equation (2.23) enriched by the influence of normally distributed noise on the input with variance σ_U^2, on the output with variance σ_Y^2 and on states with covariance matrix $\Sigma_{\mathbb{X}}$.

Besides the transition model shown on the figure 4.3, we need to specify values of variables in the first time slice. If initial conditions of input and output variables are not equal to zero, then the recalculation of the initial values of state vector is required. Moreover, state space representation has more unknown parameters than a difference equation of the same order and, as a consequence, numerical complications during identification can appear. Consequently, for providing sufficient results of identification more input data are required.

On the other hand, this representation has considerable advantages. First of all, the change of a system order does not require significant changes in the structure of a DBN. For this purpose, we have to modify the dimensions of the state vector and state matrices accordingly to the system order. This network is a 2TDBN for an arbitrary system, so it always fulfills Markov independence assumption (3.23).

If we want to use this model for identification, we have to measure the series of input and output variables. If it is possible to measure the values of state variables, they have to be also taken into account. However, mostly the state vector is immeasurable and it does not necessarily reflect real variables. If the state vector cannot be measured, then the corresponding node is considered to be hidden and searching of state matrices requires using learning algorithms for DBNs with continuous nodes that can cope with non-complete datasets.

The implementation of state space representation based structure using a static Bayesian network can be done by the unrolling of the network presented on the figure 4.3 for T/dt time slices, where T is the length of considered signals and dt is a discretization step (sampling rate). The resulting structure is presented on the figure 4.4.

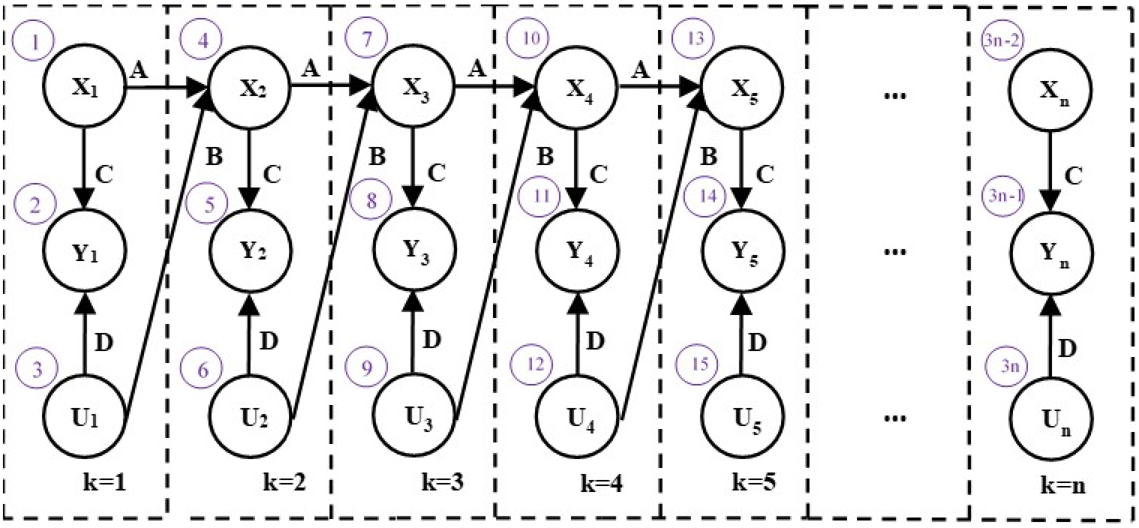


Figure 4.4: Representation of state space model using static BN

Similarly to the difference equation based model, we have to tie all corresponding parameters for different time slices to provide system identification using this type of a structure.

Only a few available software tools for Bayesian networks allow reasoning over dynamic Bayesian network with multivariate continuous nodes. The overcoming of this issue is possible by using of n univariate nodes instead of a multivariate node with n elements. The covariance matrix of a state vector in both the structure presented on the figure 4.3 and the structure presented on the figure 4.4 follows the general structure presented in (3.14). It contains the variances of each state on the main diagonal and the covariances between all pairs of state variables elsewhere. A linear Gaussian Bayesian network does not support other than linear dependences between random variables. Hence, the structure of the resulting network can contain only one from two possible cases: no direct dependence between nodes (an edge is absent) or linear dependence (an edge is present). Our previous experiments showed that the former choice lead to incorrect calculations and additional complications with the choice of weights. The latter choice is significantly better even though the state variables are not in fact independent. However, since the dependence between them is not linear, it is better to take too strong independence assumptions, regardless the fact that the information from the covariance matrix will be partially lost (all covariances between state variables are removed).

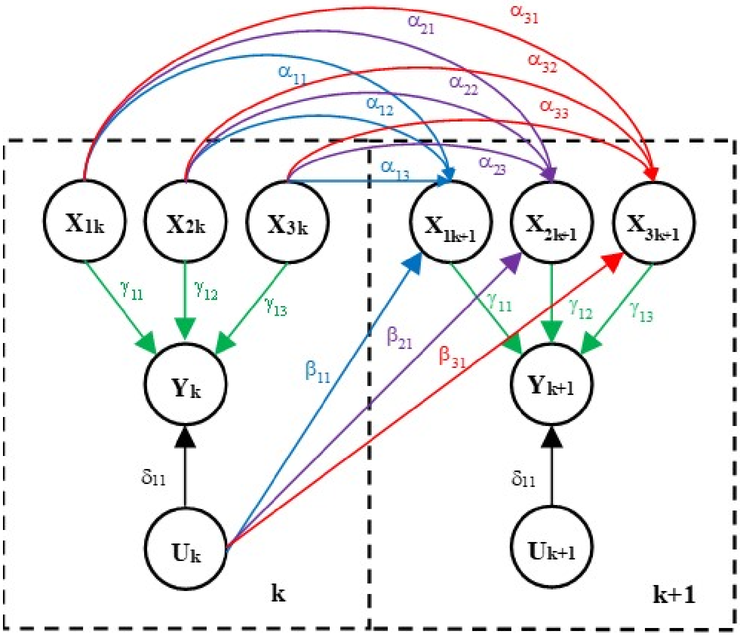


Figure 4.5: Representation of state space model using dynamic BN with independent states

The resulting structure will slightly differ for different system order n. For example, the structure of a dynamic Bayesian network for a dynamic system of the third order is presented on the figure 4.5. For this structure the following general form of state matrices was used:

$$\begin{aligned}\begin{bmatrix}x_1(k+1)\\x_2(k+1)\\x_3(k+1)\end{bmatrix} &= \begin{bmatrix}\alpha_{11} & \alpha_{12} & \alpha_{13}\\ \alpha_{21} & \alpha_{22} & \alpha_{23}\\ \alpha_{31} & \alpha_{32} & \alpha_{33}\end{bmatrix} \cdot \begin{bmatrix}x_1(k)\\x_2(k)\\x_3(k)\end{bmatrix} + \begin{bmatrix}\beta_{11}\\ \beta_{21}\\ \beta_{31}\end{bmatrix} \cdot u(k),\\ y(k) &= \begin{bmatrix}\gamma_{11} & \gamma_{21} & \gamma_{31}\end{bmatrix} \cdot \begin{bmatrix}x_1(k)\\x_2(k)\\x_3(k)\end{bmatrix} + \begin{bmatrix}\delta_{11}\end{bmatrix} \cdot u(k).\end{aligned} \tag{4.5}$$

The structure based on a static Bayesian network can be implemented using the unrolling of a dynamic Bayesian network (analogously to the structure presented on the figure 4.4). If we consider a MIMO system, then nodes corresponding to the output variable and/or input variable (which will be also multivariate in this case) have to be treated similarly to the state vector node.

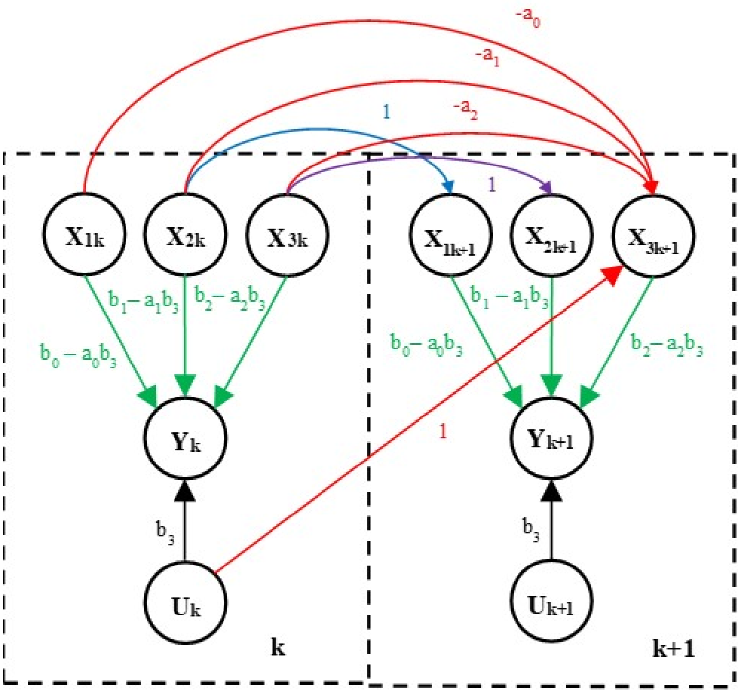


Figure 4.6: Controllable canonical form using dynamic BN

In order to improve the performance of learning algorithms, the canonical forms of the state space representation can be used. These forms are unique for each dynamic systems and therefore provide unequivocal description of the dynamics of a considered system. In addition, they explicitly contain the coefficients of a difference equation that describes a system. Moreover, in the context of system identification, the using of the canonical forms reduces the amount of unknown parameters. It is expected, that the stability of parameter computation will consequently increase as well as the precision of resulting estimates.

The amount of unknown parameters in networks based on the canonical forms of the state space representation is significantly reduced, since state matrices are partially known. For the networks presented on the figure 4.3 and on the figure 4.4 no changes in the structure of a network are required, only the weights of nodes have to be defined accordingly to a chosen canonical form, for example the controllable canonical or the observable canonical. For the Bayesian network with independent states shown on the figure 4.5 the resulting structure will slightly differ for different system order n. For example, let us consider the dynamic system of the third order. The controllable canonical form of the state space representation for the system of the third order can be expressed using the coefficients from the difference equation (2.16) as:

$$\begin{gathered}\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_0 & -a_1 & -a_2 \end{bmatrix} \cdot \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \cdot u(k), \\ y(k) = \begin{bmatrix} (b_0 - a_0 b_3) & (b_1 - a_1 b_3) & (b_2 - a_2 b_3) \end{bmatrix} \cdot \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} b_3 \end{bmatrix} \cdot u(k). \end{gathered} \tag{4.6}$$

The corresponding network structure is presented on the figure 4.6.

The observable canonical form of the state space representation (2.14) for the same system can be expressed as:

$$\begin{gathered}\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & -a_0 \\ 1 & 0 & -a_1 \\ 0 & 1 & -a_2 \end{bmatrix} \cdot \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} (b_0 - a_0 b_3) \\ (b_1 - a_1 b_3) \\ (b_2 - a_2 b_3) \end{bmatrix} \cdot u(k), \\ y(k) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} b_3 \end{bmatrix} \cdot u(k). \end{gathered} \tag{4.7}$$

The corresponding network structure is presented on the figure 4.7.

The structures based on static Bayesian networks can be implemented using the unrolling of a dynamic Bayesian network (analogously to the structure presented on the figure 4.4).

We can also construct other canonical form of the state space representation (e.g. modal form) using similar approach.

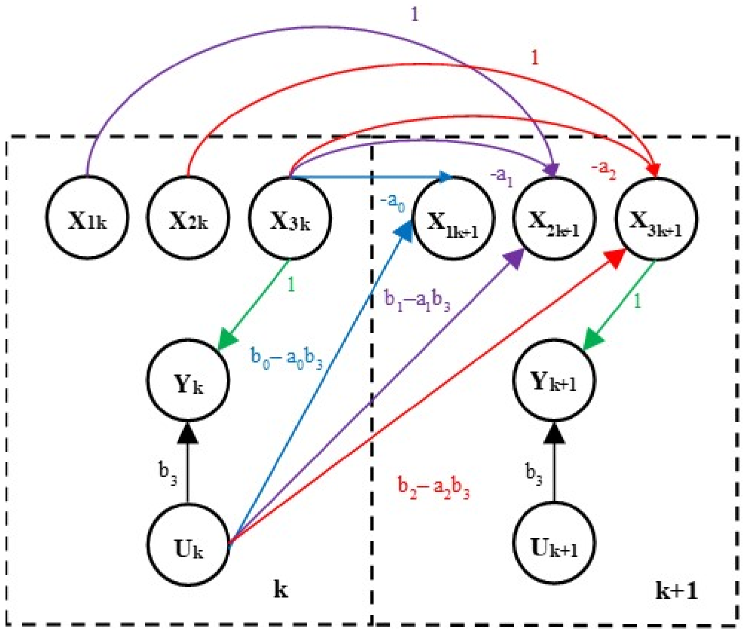


Figure 4.7: Observable canonical form using dynamic BN

All above-mentioned structures are not task-specific and therefore can be used for any dynamic system. In addition, we can implement any arbitrary structure of a state space model. The information about dependencies and independencies between random variables can be obtained from expert knowledge (for example, if we construct the model of a technological process and state variables correspond to physical variables). Unlike all structures presented earlier in this section, the structure of such network would be task-specific, but the including of all a priori knowledge will reduce the amount of unknown parameters and as a consequence, the stability of computations and the precision of obtained estimates are expected to increase.

4.3 Bayesian network based state observers

Bayesian networks can be also used as state observers, i.e. they can be used for state estimation. Bayesian network based state observers are not in the scope of this thesis, therefore their structures and basic steps of their implementation are proposed briefly. However, since this task is closely related to the task of modelling and identification of dynamic systems using state space representation, this discussion cannot be omitted.

Solving of the state estimation task often requires knowing of the parameters of a considered system and the measurements of input and output signals. The parameters of a system can be obtained either from identification procedure or from expert knowledge regarding a considered system.

The values of state matrices should be defined as weights in a Bayesian network and the values of input and output variables in j time steps are added as evidence to corresponding nodes. The values of state variables in a time step k can be obtained from the Bayesian network by the marginalization of the node corresponding to the state vector (or several nodes for structures with univariate nodes for each state variable). Depending on the values of j and k we can distinguish among three different problems the state estimation can be provided for: filtering ($k = j$), smoothing ($k < j$) or prediction ($k > j$) [2]. State estimation can be provided using any structure from those proposed in the section 4.2.

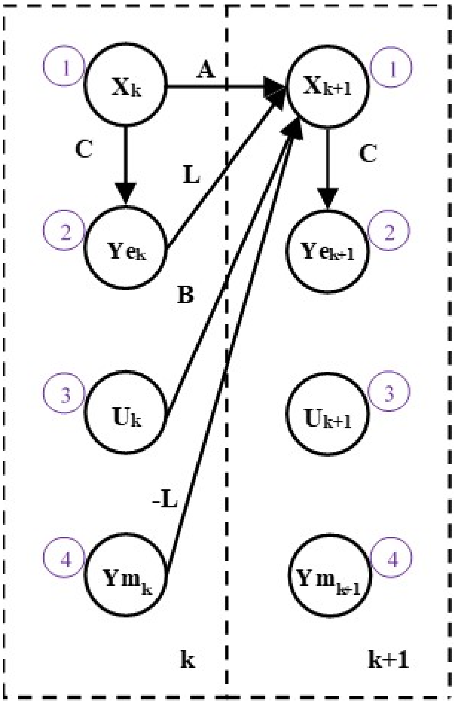


Figure 4.8: Bayesian network based Luenberger state observer

Bayesian networks can be used for state estimation in combination with popular state observers, e.g. Luenberger observer or Kalman filter (refer to the section 2.4). The structure that corresponds to a Bayesian network based Luenberger observer is presented on the figure 4.8. This structure combines a Bayesian network from the figure 4.3 and Luenberger state observer (2.33). The node Um_k represents the measurement of the input signal. The output variable is presented in this structure by two different nodes: Ye_k for the estimated value of the output and Ym_k for the measured value of the output. The matrix $\mathbb{L}$ is chosen according to desired dynamics of estimation error (refer to the section 2.4 for more details). The state vector can be calculated from the Bayesian network presented on the figure 4.8 using the marginalization of the node X_k.

The structure that corresponds to a Bayesian network based Kalman filter (refer to the sections 1.1 and 2.4 for more details on Kalman filter) is presented on the figure 4.9. This structure combines a Bayesian network from the figure 4.3 and Kalman filter (2.39) or (2.40). The node U_k represents the measurement of the input signal. The output variable is presented in this structure by two different nodes: Yc_k for the corrected value of the output and Ym_k for the measured value of the output. The state vector is also represented by two nodes: Xp_k for the predicted value of the state vector and Xc_k for the corrected value of the state vector. The matrix $\mathbb{K}$ can be calculated from the matrix $\mathbb{P}$ obtained by the solving of Ricatti equation (2.41) and then the structure on the figure 4.9 corresponds to a Bayesian network based steady-state Kalman filter (2.40). The corrected state vector can be obtained from this structure by the marginalization of the node Xc_k. If we want to update the value of the matrix $\mathbb{K}$ iteratively, i.e. use the traditional Kalman filter (2.39), then the value of matrix $\mathbb{K}$ has to be calculated at each time step before the marginalization of the node Xc_k.

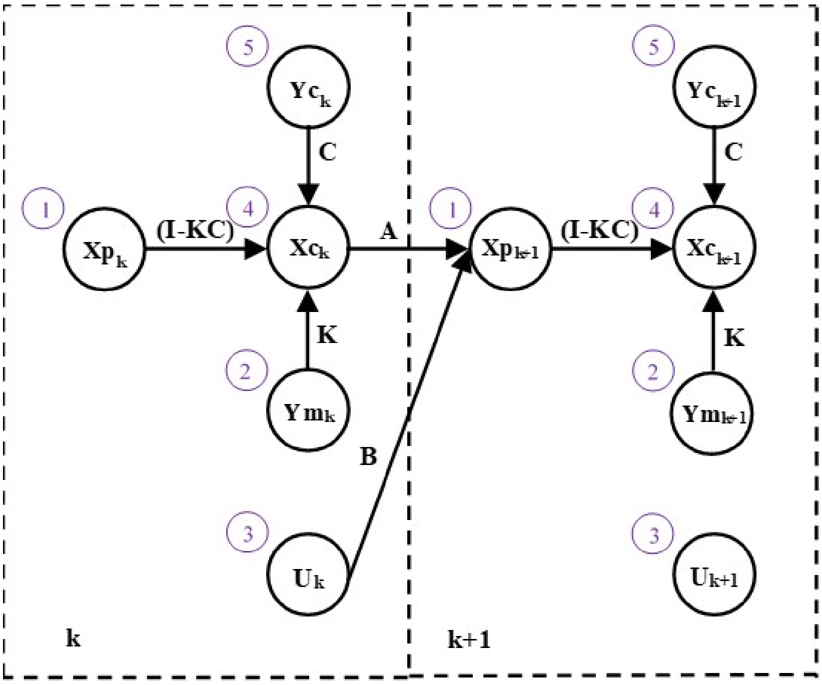


Figure 4.9: Bayesian network based Kalman filter

5 Identification of dynamic systems using Bayesian networks

The goal of identification is to find the description of a considered dynamic system that reflects its behavior with acceptable precision. In the case of parametric identification, which is considered in this thesis, we assume that the type of a model is known in advance and hence, the identification task reduces to the estimation of unknown parameters of a known model. In this thesis, we propose to use Bayesian networks for this task.

The general methodology of Bayesian network based system identification (BNSI) is presented in the section 5.1. This methodology is not software-specific, and hence, any appropriate software package can be used for its implementation.

The structure of a Bayesian network used in practical experiments, corresponds to a dynamic Bayesian network based state space representation of dynamic systems (refer to the section 4.2). This choice was made due to several important reasons. Firstly, the identification of dynamic systems using the model based on a difference equation (refer to the section 4.1) leads to the estimation of parameters using MLE (well-studied least square method). Secondly, according to provided review over identification methods in the section 1.1, only several tools are available for the identification of state space models (i.e. subspace methods and EKF). Therefore, A BNSI can be a perspective new tool for this purpose. And finally, the chosen software package for the implementation of BNSI (BNT for MATLAB) supports only 2TDBN, therefore it is not possible to implement a difference equation based model for the dynamic systems of the second or higher order using DBNs in BNT.

The applicability of the proposed approach was validated on the plenty of practical experiments. The first set of experiments aimed to explore the learning procedure that would be the most efficient for Bayesian network based system identification. The efficiency was assessed using both the precision of obtained estimates and numerical properties of a corresponding learning procedure (i.e. computation time, required amount of iterations and numerical stability). Searching for the optimal learning procedure and the studying of the influence of different tuning parameters on its efficiency was provided on simulated deterministic systems. The proposed procedure was consequently verified on dynamic systems enriched with simulated stochastic component and the obtained results were compared with commonly used method for state space identification of dynamic systems (N4SID). The proposed order selection procedure was also studied on simulated stochastic dynamic systems and the obtained results were compared with commonly used order selection criterion (AIC).

Finally, the proposed approach to system identification was applied for responses of two real dynamic systems: the aperiodic system and the oscillate system. The identification procedure was applied on these responses using different discretization steps. The applicability of scoring functions used in structure learning of Bayesian networks for order selection was verified. Obtained results of identification were compared with N4SID and suggested orders were compared with AIC.

The section 5.2 describes how all simulations and experiments were prepared, provided and evaluated. Searching for the optimal learning procedure using simulated deterministic responses is described in the sections 5.3, 5.4 and 5.5. The influence of introduction the expert knowledge into a Bayesian network in the form of variances of considered variables is explored in the section 5.3. On the basis of these experiments, the most precise setting was chosen. Further experiments explore the influence of including the partial knowledge regarding parameters (section 5.4) and the influence of the type of distribution of initial values of parameters used by the EM algorithm (section 5.5) on the precision of obtained estimates and the duration of identification procedure.

The efficiency of Bayesian networks in the identification of stochastic dynamic systems is explored in the section 5.6. In the section 5.7 the proposed order selection approach is presented and verified on the simulated stochastic responses. Verification of proposed approaches on datasets obtained from real dynamic systems is provided in the chapter 6.

5.1 General methodology of Bayesian network based system identification

The implementation of Bayesian networks for the identification of dynamic systems can be provided using one of settings defined by the combination of several design choices. The most influential design choices are the structure of a Bayesian network and the type of used learning algorithm.

The choice of a setting depends on the task the identification procedure is provided for, on the demands made on algorithm (e.g. computation time, precision) and on the capabilities of a software package chosen to carry out the identification task.

The identification of dynamic systems using Bayesian networks can be provided using the following steps:

1. Definition of a Bayesian network structure that describes the behavior of a dynamic system
2. Including of any expert knowledge available for a considered system (e.g. variances of random variables, independencies between them or partial knowledge regarding the parameters of a system)
3. Introduction of identification measurement into a considered model (providing evidence to a Bayesian network)
4. Identification of a dynamic system (learning parameters of a Bayesian network)

A setting used for Bayesian network based system identification is dependable on several design choices:

1. **Choice of the structure of a considered network** (refer to the chapter 4). The structure can be based on a difference equation or on a state space representation. Using of the canonical forms of a state space representation considerably reduces the amount of unknown parameters, especially for higher order systems. It positively influences the numerical properties of learning algorithm and reduces the amount of input data required for obtaining successful results.
2. **Choice of the type of a used network: static or dynamic**. If the static network is used, then the structure chosen in the previous step has to be unrolled for T/dt time slices, where T is the duration of identification measurement and dt is a discretization step (refer to the chapter 4 for the examples of the unrolling of a DBN). The choice of a network type depends on the capabilities of used software package. Some packages do not support dynamic Bayesian networks and hence unrolling the network is the only option. Also, the amount of supported inference algorithms (required for learning networks with a non-complete dataset) for static networks is often higher than for dynamic networks. Therefore, the static network can be used to employ them for the identification task. On the other hand, inference algorithms for

dynamic systems are often faster and require less main memory (unless an algorithm requires the unrolling of the network into T/dt time slices). Moreover, in dynamic Bayesian network based models the parameter tying (the use of the same parameters for the same variables in different time slices) is implemented as a part of the definition of a network structure. In static networks, the parameters of a network have to be tied manually.

3. **Choice of the type of required estimates**. If the goal of identification is to find the distributions over unknown parameters, then the procedures for Bayesian estimation have to be chosen. On the other hand, if we are interested in the point estimates of unknown parameters, then we search for maximum likelihood estimate (MLE). The latter option has more support in available software packages.

4. **Choice of the type of inference algorithm for networks with non-complete data**. Non-completeness of data may appear if there are missing data (variables that could not be measured occasionally) or hidden nodes (variables that are unmeasurable, e.g. state variables). In this situation, it is not possible to apply the learning procedure until the dataset is complete. The algorithm that can cope with this issue is an Expectation Maximization (EM) algorithm, in which the missing values are computed in E step given the current estimates of parameters and then the parameters are computed in M step given the current estimates of missing values. The chosen inference algorithm provides the estimation in E step, whereas the M step is provided using classical maximum likelihood estimation.

The resulting setting is often modular, i.e. the same learning algorithm can be used for different network structures or the same network structure can be learned using different algorithms (e.g. we can use different inference algorithm in E step of EM algorithm). This property does not hold in specific cases, e.g. we cannot use the algorithm designed for dynamic Bayesian networks for learning of a static Bayesian network. However, in such cases, we can always find the equivalent algorithm for static case.

5.2 Preparation of experiments and their evaluation

The experiments provided to study Bayesian network based system identification (BNSI) aimed to explore the applicability of different settings, the influence of different tuning parameters on the learning procedure and the consequences of including partial knowledge regarding a considered system. For the generalizability of obtained results it was necessary to apply system identification procedure on different dynamic systems with different length of input signals. The exploration of the most suitable learning scenario and the influence of different parameters was provided for deterministic systems, whereas the precision of obtained results and the order selection methodology were assessed on stochastic dynamic systems.

The datasets obtained from identification measurement provided on two dynamic systems were used for the verification of the applicability of the proposed approach to system identification and the proposed approach to order selection.

5.2.1 Implementation of system identification procedure

Experiments described in this chapter were implemented in MATLAB [181] using BNT [180]. The general algorithm for parameter learning in Bayesian networks using BNT is presented on the figure 5.1. The network structures used in system identification experiments are shown on the figure 5.2, where the red circles represent the numbers of corresponding nodes in the structure of a Bayesian network. The numeration strategy for nodes in BNT yields that each parent node must have smaller number than its child nodes. Both structures fulfill this requirement.

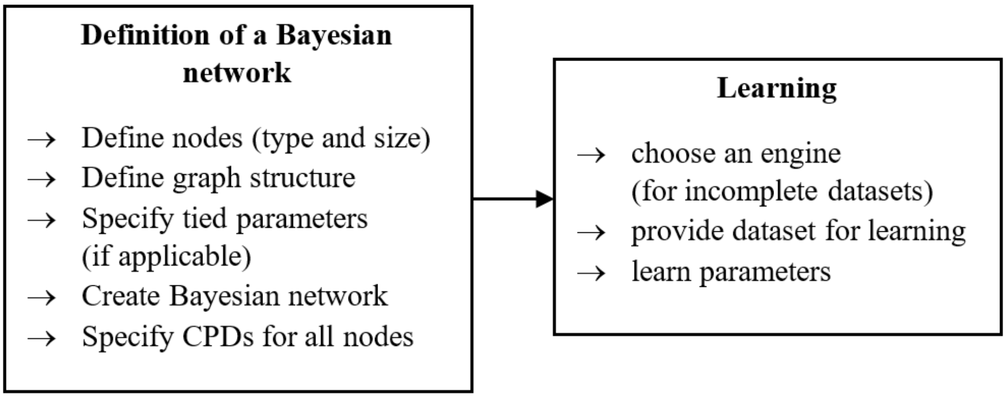
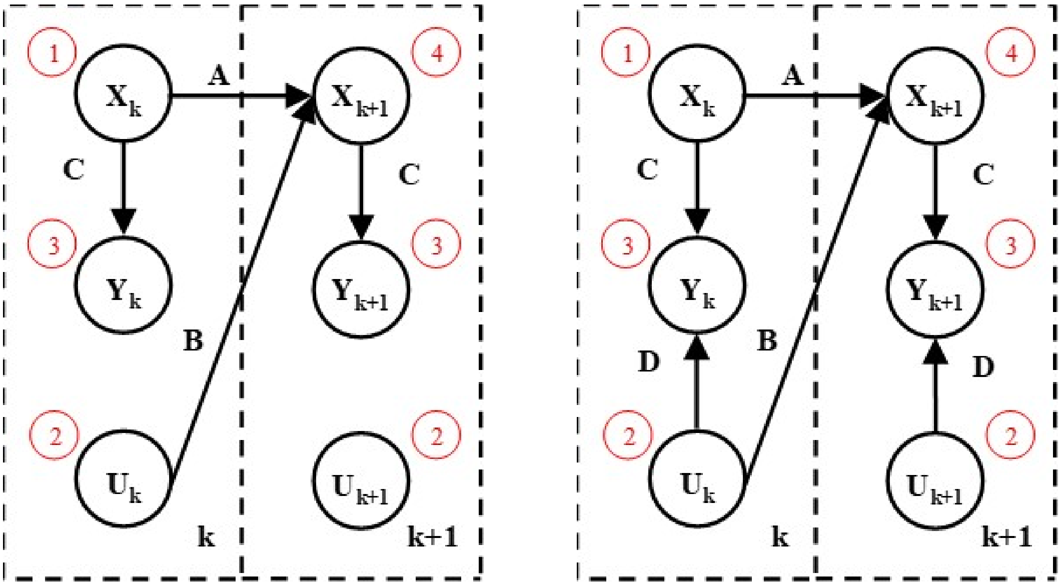

Definition of a Bayesian network
→ Define nodes (type and size)
→ Define graph structure
→ Specify tied parameters (if applicable)
→ Create Bayesian network
→ Specify CPDs for all nodes
Learning
→ choose an engine (for incomplete datasets)
→ provide dataset for learning
→ learn parameters

Figure 5.1: General algorithm of parameter learning in BNT

The majority of considered systems do not have direct feedthrough (state matrix $\mathbb{D}$ is equal to zero) and for such cases the structure from the figure 5.2a is sufficient. However, the identification of a dynamic system with direct feedthrough was also included into provided experiments for the generalizability of obtained results. For this system the structure from the figure 5.2b was used. It was not possible to unify identification trials by using only the latter structure due to the limitations of BNT (it would not be possible to fix the value $\mathbb{D} = 0$ during learning for systems without direct feedthrough without fixing the values in matrix $\mathbb{C}$).

Implementation of both structures is almost identical, the changes that have to be done for systems with direct feedthrough will be described in comments.



(a) Systems without direct feedthrough

(b) Systems with direct feedthrough

Figure 5.2: Bayesian network structures used for identification

The definition of Bayesian networks in experiments was provided according to the algorithm presented on the figure 5.1. First, we have to define the type of all nodes (discrete or continuous) and their size (scalar or vector). We consider SISO dynamic systems and hence variables U and Y are scalars, the size of state vector is equal to the order of considered dynamic system, all nodes of our interest are continuous:

```
n = 3; %the order of a considered dynamic system
N = 3; %the number of nodes
X = 1; U = 2; Y = 3; %alliases for all nodes

%specify the size and type of each node
discrete_nodes = [ ]; %all nodes are continuous
node_sizes = [n 1 1]; %X is a vector, U and Y are scalars
observed_nodes = [U Y]; %input U and output Y are observed at each time slice
```

The definition of structure in static Bayesian networks is provided by a $N \times N$ matrix (where N is the number of nodes in a network) that consists of ones and zeros. Each unity corresponds to a directed edge that goes from a node with a number equal to the number of a row to the node with a number equal to the number of a column. In dynamic Bayesian networks the structure of a graph is described by two $N \times N$ matrices. One of them represents all edges inside the same time slice and another one represents all edges that goes from nodes in one time slice to nodes in the next time slice:

```
%intra - connections between nodes in the same slice
intra = zeros(N);
intra(X, Y) = 1; %arc from X(k) to Y(k)
%intra(U, Y) = 1; %this row has to be added for systems with direct feedthrough

%inter - connections between nodes in different time slices
inter = zeros(N);
inter(X, X) = 1; %arc from X(k) to X(k+1)
inter(U, X) = 1; %arc from U(k) to X(k+1)
```

Also, we have to define, which parameters are tied (a parameter is called tied, if it does not change for different time instances). For dynamic Bayesian networks tying of the parameters for the same variables in different time slices is implemented automatically, however, we have to define, which nodes have different distributions in the first time slice and the consequent slices. In the considered structure such node is a state vector. In the first time slice it has no parent nodes and the values of its elements correspond to the values of initial conditions. In further time slices the value of state vector depends on the value of input and state vector in the previous time slice. Due to this reason, state vectors in the structures on the figure 5.2a and on the figure 5.2b have two different numbers in different time slices. Remaining nodes (input and output variables) stay tied, since they do not change:

```
eclass1 = [1 2 3]; %CPDs of U and Y stay the same at each time slice
eclass2 = [4 2 3]; %CPD of X in the second time slice is different
```

Now we can create a dynamic Bayesian network:

```
bnet = mk_dbn(intra, inter, node_sizes, 'discrete', discrete_nodes, ...
'observed', observed_nodes, 'eclass1', eclass1, 'eclass2', eclass2);
```

After this step, we have to specify CPDs for all nodes. In BNT the only supported type of distribution that can be used for continuous nodes is normal distribution. Other software packages mostly use the same representation of continuous nodes or support discretization of continuous nodes and consequently treat them as discrete. Normal distribution for continuous nodes is in general defined in BNT as:

```
bnet.CPD{1} = gaussian_CPD(bnet, 1);
bnet.CPD{2} = gaussian_CPD(bnet, 2);
bnet.CPD{3} = gaussian_CPD(bnet, 3);
bnet.CPD{4} = gaussian_CPD(bnet, 4);
```

This definition implies that means, covariance matrices (or variances in the case of scalar nodes) and weights are sampled from the standard normal distribution. Their values can be defined using arguments 'mean', 'cov' and 'weights' respectively. If we define these values and start the learning procedure, they will be taken as initial values of parameters and will be updated by EM learning procedure. We can avoid updating of parameters by using appropriate binary argument: 'clamp_mean', 'clamp_cov' or 'clamp_weights'. In addition, we can set the covariance matrix to a diagonal form by setting of argument 'cov_type' to the value 'diag' (the default value of this parameter is 'full', which corresponds to the unconstrained form of the covariance matrix).

The values of weights correspond to the parameters of a dynamic system that are unknown during system identification. For each node, 'weights' is represented by a vector (for univariate nodes) or a matrix (for a multivariate nodes). The parameters associated with a parent node with smaller number will appear in the columns with a smaller number. When the 'clamp_weights' argument is used, the values of weights associated with all parents have to be specified, since it is not possible in BNT to set the values of weights partially.

The next step is to define inference engine for incomplete datasets. Since state vector is in most cases unmeasurable, dataset in the identification of considered systems will be technically incomplete (state vector is represented by a hidden node). Therefore, we have to choose appropriate engine for the calculation of state vector in the E step of EM algorithm. There are several inference algorithms for dynamic Bayesian networks. From preparatory experiments we found out, that only three of them can be used for learning of networks with the structure corresponding to the state space representation: junction tree algorithm, unrolled junction tree algorithm and Pearl algorithm. Unfortunately, Pearl algorithm provided considerably worse estimates than both of junction tree based algorithms. Therefore, it was not used in experiments described in this thesis. The Kalman inference algorithm that is available in BNT was not used as well, since it is designed for state space models without external input.

Obviously, junction tree algorithm and unrolled junction tree algorithm use the same inference method with one difference: unrolled junction tree unrolls a dynamic network and provides inference in a static network, whereas junction tree provides inference on pairs of neighboring slices in a dynamic network. The latter algorithm is therefore faster, but the former one is more stable in computations. However, the usability of unrolled junction tree depends on the amount of time slices in considered network and on the capacity of main memory of a computer used for learning. The inference algorithm is defined as:

```
%engine = jtree_dbn_inf_engine(bnet);
engine = jtree_unrolled_dbn_inf_engine(bnet, len);
```

Variable *len* is the length of a dataset, i.e. the amount of time slices for which the engine has to unroll a dynamic network. The datasets for learning were loaded from corresponding external files. Input data used for identification are stored in a cell array with the following structure:

```
data = cell(1, 1);
data{1} = cell(N, len);

for i = 1:len
data{1}([U Y], i) = num2cell([data_u(i); data_y(i)]);
end
```

Now we can apply the EM algorithm for the learning of parameters in a Bayesian network:

```
[bnet2, loglike, engine2] = learn_params_dbn_em(engine, data);
```

The learning algorithm has several arguments, which influence its behavior. The most crucial ones are threshold (‘thresh’) for stopping the EM (the default value is 0.001) and maximum number of iterations (‘max_iter’) that will be performed, if the algorithm will not reach defined threshold (the default value is 10 for static and 100 for dynamic Bayesian networks). The threshold is a bound for the fractional change in the value of log likelihood function. Reaching of this threshold is considered by the algorithm as reaching the optimum of log likelihood function. Preliminary experiments have shown that decreasing of this argument does not have significant influence on the precision of obtained results and prolong an identification trial. Therefore, in all experiments, the threshold was not changed. The experiments have also shown that in some settings the learning algorithm does not have enough time to converge and therefore, the maximum number of iterations was set to the value 1000.

After parameter learning, it is important to correctly interpret obtained results. The values of parameters are stored in the weights of each node. To get the access to this information we have to break the structure's privacy by the following code:

```
%Access node structures
s1 = struct(bnet2.CPD{1});
s2 = struct(bnet2.CPD{2});
s3 = struct(bnet2.CPD{3});
s4 = struct(bnet2.CPD{4});
```

Assuming that all means are set to zero values during identification (which was the case for all experiments described in this thesis), the state matrices can be obtained as:

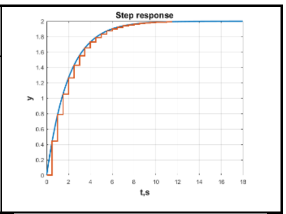
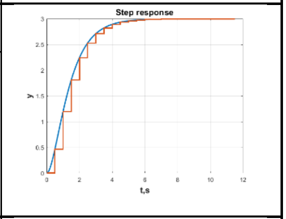
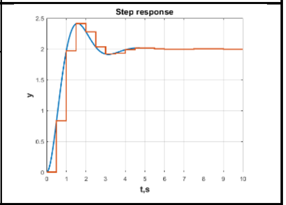
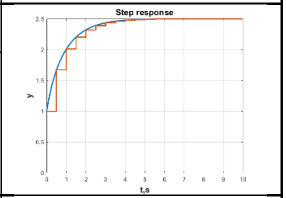
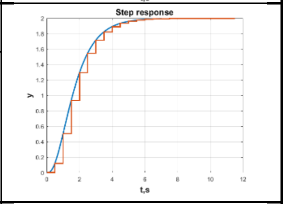
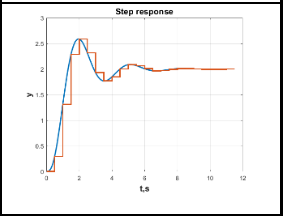
```
A = s4.weights(:, 1:n);
B = s4.weights(:, end);
C = s3.weights(:, 1:n);
D = 0; %D = s3.weights(:, end); %for systems with direct feedthrough
```

Further evaluation can be provided using common tools for analysis of systems, e.g. the functions from Control System Toolbox in MATLAB.

5.2.2 Experiment setups

For analysis of different settings and different learning scenarios, the behavior of six dynamic systems was simulated in MATLAB, see table 5.1. Chosen types of transfer functions for considered systems correspond to the most frequently used types for system identification of linear dynamic systems. The parameters of transfer function were chosen in such way that they have similar time required to achieve the steady state and similar time constants. As a consequence, the unified input signals with unified discretization step can be used for their simulations. It was important to choose the efficient value of discretization step to assure identifiability of simulated responses. The Nyquist-Shannon sampling theorem [227] suggests that the minimum value of sampling frequency is twice the highest frequency of a dynamic system. The value of discretization step $dt = 0.5s$ satisfies this theorem for all considered dynamic systems in the table 5.1, except the system of the first order. However, for

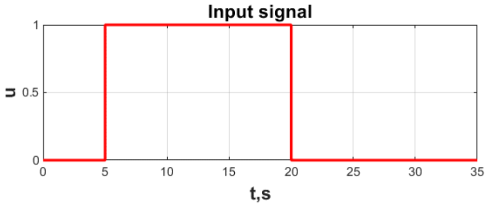
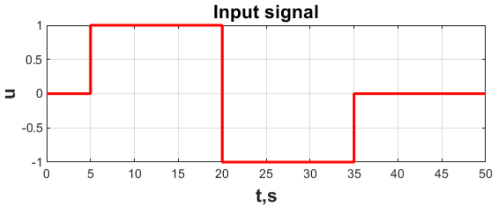
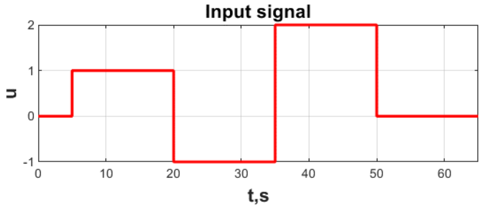
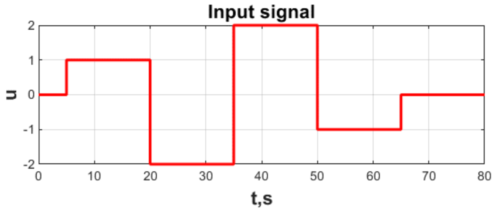
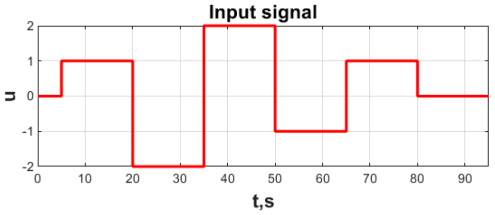
Table 5.1: Considered dynamic systems

Transfer function	Transfer function (discrete)	Step response
The system of the **first** order		
Transfer function (continuous): $G(s) = \frac{2}{2s+1}$	Transfer function (discrete, $dt = 0.5\ s$): $G(z) = \frac{0.4424}{z - 0.7788}$	
The system of the **second** order (**aperiodic**)		
Transfer function: $G(s) = \frac{6}{s^2 + 3s + 2}$	Transfer function (discrete, $dt = 0.5\ s$): $G(z) = \frac{0.4645z + 0.2817}{z^2 - 0.9744z + 0.2231}$	
The system of the **second** order (**oscillate**)		
Transfer function: $G(s) = \frac{10}{s^2 + 2s + 5}$	Transfer function (discrete, $dt = 0.5\ s$): $G(z) = \frac{0.8342z + 0.5907}{z^2 - 0.6554z + 0.3679}$	
The system of the **second** order (**with direct feedthrough**)		
Transfer function: $G(s) = \frac{s^2 + 7s + 10}{s^2 + 5s + 4}$	Transfer function (discrete, $dt = 0.5\ s$): $G(z) = \frac{z^2 - 0.07313z - 0.07632}{z^2 - 0.7419z + 0.08208}$	
The system of the **third** order (**aperiodic**)		
Transfer function: $G(s) = \frac{12}{s^3 + 6s^2 + 11s + 6}$	Transfer function (discrete, $dt = 0.5\ s$): $G(z) = \frac{0.1218z^2 + 0.2374z + 0.02718}{z^3 - 1.198z^2 + 0.4406z - 0.04979}$	
The system of the **third** order (**oscillate**)		
Transfer function: $G(s) = \frac{6}{0.25s^3 + s^2 + 1.9s + 3}$	Transfer function (discrete, $dt = 0.5\ s$): $G(z) = \frac{0.2959z^2 + 0.7052z + 0.11}{z^3 - 1.062z^2 + 0.7529z - 0.1353}$	

this system slightly lower value of a discretization step than the one suggested by the Nyquist-Shannon theorem did not cause problems during identification due to the simple structure of a considered model.

The further choice was the type of input signal for considered dynamic systems. One of the most common types of input signals used for system identification in practice is the unity step, since the response on this signal is very informative. The

Table 5.2: Types of used input signals

Type	Properties	Plot
‘1 step’	Length: $T = 35\ s$ Discretization step: $dt = 0.5\ s$ The amount of points in a dataset: 71	
‘2 steps’	Length: $T = 50\ s$ Discretization step: $dt = 0.5\ s$ The amount of points in a dataset: 101	
‘3 steps’	Length: $T = 65\ s$ Discretization step: $dt = 0.5\ s$ The amount of points in a dataset: 131	
‘4 steps’	Length: $T = 80\ s$ Discretization step: $dt = 0.5\ s$ The amount of points in a dataset: 161	
‘5 steps’	Length: $T = 95\ s$ Discretization step: $dt = 0.5\ s$ The amount of points in a dataset: 191	

prominent modifications of this signal are non-unity step and the sequence of steps with different height. Since it was required to explore the influence of increasing the amount of input data on results of identification, the latter type of signal was chosen with various amount of steps, see table 5.2. It is worth to point out, that increasing of dataset by decreasing a discretization step would not be informative, since it can lead to additional numerical issues due to the small values of resulting parameters.

The deterministic responses (without the influence of noise) of dynamic systems described in the table 5.1 on input signals described in the table 5.2 were simulated in Simulink with discretization step $dt = 0.5s$. The resulting 30 datasets were used for identification experiments described in the sections 5.3, 5.4 and 5.5.

For experiments described in the sections 5.6 and 5.7, the behavior of stochastic dynamic systems had to be simulated. The models used for previous simulations were enriched by a random noise (sampled from normal distribution with zero mean), which was added to the output variable and to each variable in the state vector. The chosen values of the variance of random noise were $\sigma_{\epsilon}^2 = 0, 0.01, 0.02, 0.05, 0.1$. The simulation procedure with mentioned settings lead to 150 datasets used in these experiments.

5.2.3 Evaluation of results

Since the result of parameter learning using EM algorithm strongly depends on the initial values of parameter estimates, it was required to provide sufficient amount of identification trials to get reasonable results of Bayesian network based identification. Therefore, for each setting considered in this thesis, the bunch of 100 identification trials were provided. For preserving the repeatability of conducted experiments, the random number generator used for random sampling of initial parameter values was seeded by an integer corresponding to the number of identification trial.

The results that were considered successful for each type of a dynamic system and each type of input signal are presented on the figure 5.3. Magenta crosses represent the output signal used for identification, the blue line corresponds to the response of the identified model of a dynamic system (a dynamic system with parameters equal to those obtained from estimation).

Due to the numerous amount of conducted experiments it was necessary to design the evaluation procedure that can be informative. For each bunch of identification trials, the values of resulting state matrices were stored to the external file for further evaluation of obtained estimates. In addition, information about the duration of learning procedure and the amount of iterations required for convergence of EM algorithm for each trial was stored for further evaluation of numerical properties of each considered setting.

Some of identification trials had numerical problems caused by singular matrices or matrices that are close to singularity. It often resulted in the inappropriate estimates of parameters (non-stable descriptions or zero-valued state matrices). In some cases, however, the algorithm did not manage to overcome the numerical issues and failed. The information about the amount of fails for each bunch of identification trials was therefore stored to external file as well.

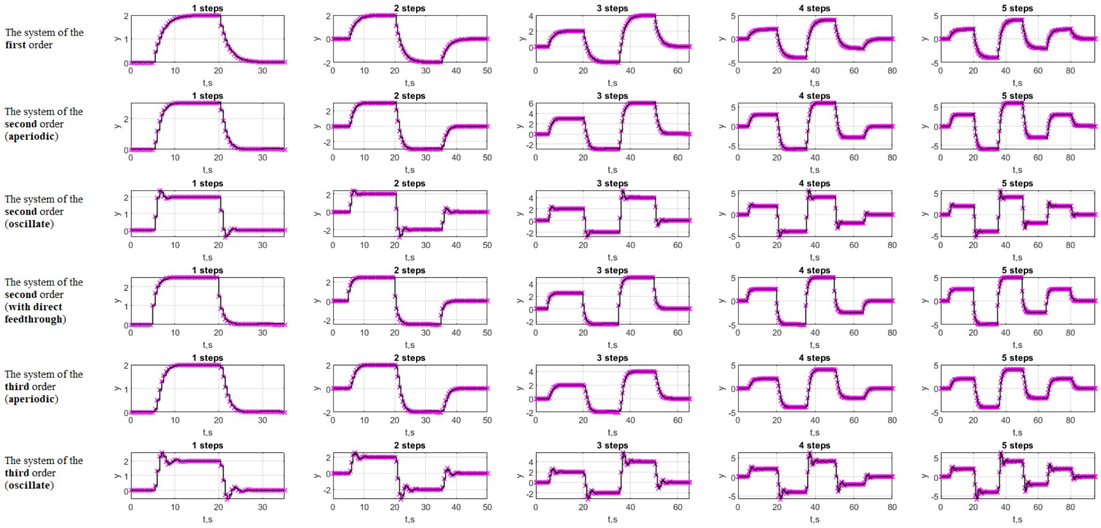


Figure 5.3: Successful results of identification for deterministic systems

The evaluation of results was consequently provided for each bunch of identification experiments. The mean squared error (MSE) was chosen as the cost function used for assessing the goodness of fit:

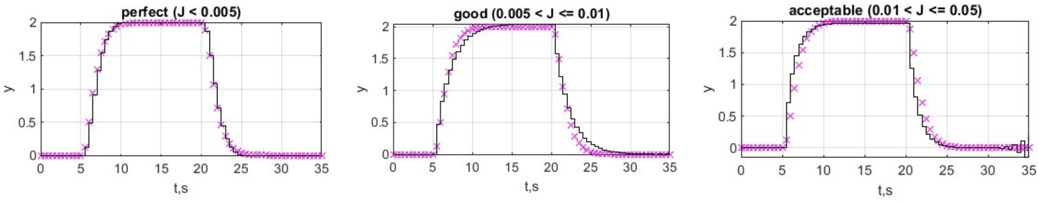
$$J = \frac{1}{m_{id}} \cdot \sum_{i=1}^{m_{id}} (y_{id_i} - y_{sim_i})^2. \tag{5.1}$$

In the equation (5.1), m_{id} is the length of a dataset used for identification, y_{sim_i} are simulated values of output used for identification, y_{id_i} are values of the response of the identified model of a dynamic system. The cost function therefore represents the average squared residual of the response of identified system from the original dataset.

In the experiments provided on deterministic dynamic systems (sections 5.3, 5.4 and 5.5), each of identification trials was assigned to one of four classes according to the value of the cost function:

- Class 1: $J \leq 0.005$;
- Class 2: $0.005 < J \leq 0.01$;
- Class 3: $0.01 < J \leq 0.05$;
- Class 4: $J > 0.05$.

The bounds of each class was chosen by visual comparison of resulting responses with original data. The first class corresponds to the perfect fit of identified system



(a) Perfect fit (b) Good fit (c) Acceptable fit

Figure 5.4: Examples of results from classes 1 - 3

to the original data, refer to the figure 5.4a for the example of such result. The second class has slightly worse goodness of fit, which, however, can be still considered as good, refer to the figure 5.4b for the example of such result. The residuals for the third class are more distinguishable, but the response of identified system is still relatively close to the original dataset and therefore these results are considered acceptable, refer to the figure 5.4c for the example of such result. The fourth class contains unsuccessful results of identification that include zero-valued parameters and other unreasonable results. These results appear when the EM algorithm faces numerical issues or gets stuck in local optima of log likelihood function, refer to the figure 5.5 for the examples of such results.

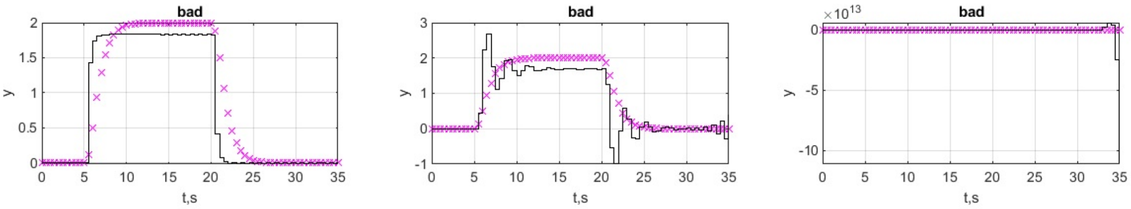
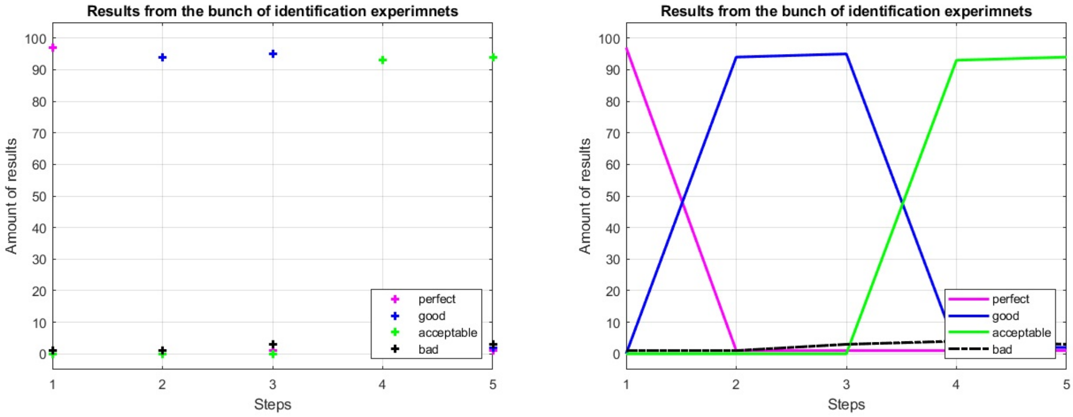


Figure 5.5: Examples of results from class 4

The amounts of results in each class were calculated for each bunch of identification trials. Plots were chosen as a tool for their appropriate representation. The horizontal axis represents 5 bunches of identification trials with different type of input signal ('1 step', '2 steps', '3 steps', '4 steps', '5 steps') and is used to assess the influence of increasing the amount of input data. The vertical axis represents the amount of corresponding results. The figure 5.6a represents the results of such evaluation. The magenta crosses correspond to the amount of results in class 1 (perfect fit), the blue crosses correspond to the amount of results in class 2 (good fit), the green crosses correspond to the amount of results in class 3 (acceptable fit), the black crosses correspond to the amount of results in class 4 (bad fit).

To increase the readability of plots, it was decided to connect points on the plot on the figure 5.6a with lines. The results classified as bad were connected with a dash line to distinguish them from successful results (perfect, good and acceptable), see figure 5.6b for the example of a resulting plot.



(a) Original plot

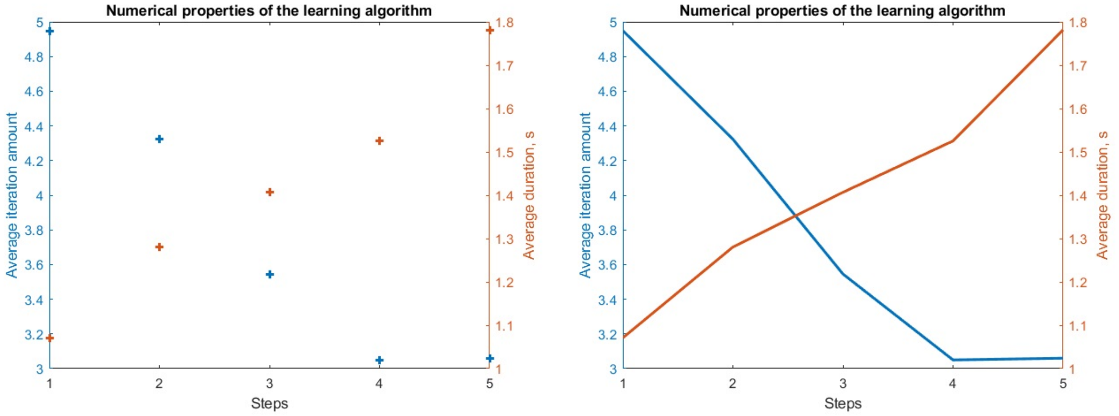
(b) Modified plot

Figure 5.6: Evaluation of results from a bunch of identification trials

For stochastic dynamic systems, the proposed evaluation procedure is not applicable, in particular for systems with higher value of noise, since it is unlikely to get the value of cost function smaller than the value of noise. Therefore, the bounds of classes for stochastic dynamic systems are modified with respect to the noise variance σ_ϵ^2:

- Class 1: $J \leq \sigma_\epsilon^2$;
- Class 2: $\sigma_\epsilon^2 < J \leq 2 \cdot \sigma_\epsilon^2$;
- Class 3: $2 \cdot \sigma_\epsilon^2 < J \leq 10 \cdot \sigma_\epsilon^2$;
- Class 4: $J > 10 \cdot \sigma_\epsilon^2$.

Further properties of different learning settings that were evaluated for each bunch of identification trials were the average duration of learning procedure and the average amount of iterations required for EM algorithm to converge. These values are also represented on plots for each experiment. The horizontal axis corresponds to the different length of input signal used in simulations ('1 step', '2 steps', '3 steps', '4 steps', '5 steps'). There are two vertical axes on plots (since represented values are in different range): the left one represents the average amount of iterations and the right one represents the average duration of learning procedure in a corresponding bunch (in seconds). Analogously to the previous case, the representation with points is badly readable, see figure 5.7a. Therefore, it was decided to connect the points with lines, see figure 5.7b for the example of a resulting plot.



(a) Original plot

(b) Modified plot

Figure 5.7: Evaluation of the numerical properties of a learning algorithm

5.3 Analysis of different learning scenarios

The conditional probability distributions of nodes in a considered Bayesian network structure for systems with direct feedthrough were already defined in (4.4). For systems without direct feedthrough the direct influence of input signal on the output signal should be removed:

$$\begin{gathered}\mathbb{X}_{k+1}|\mathbb{X}_k, U_k \sim \mathcal{N}(\mu_{\mathbb{X}} + \mathbb{A}\cdot\mathbb{X}_k + \mathbb{B}\cdot U_k, \Sigma_{\mathbb{X}_k}),\\ Y_k|\mathbb{X}_k \sim \mathcal{N}(\mu_Y + \mathbb{C}\cdot\mathbb{X}_k, \sigma_Y^2),\\ U_k \sim \mathcal{N}(\mu_U, \sigma_U^2).\end{gathered} \tag{5.2}$$

During system identification procedure, the values of input and output signals are introduced to a Bayesian network as evidence. The learning task is to find the unknown state matrices $\mathbb{A}, \mathbb{B}, \mathbb{C}$ and $\mathbb{D}$. In addition, there are unknown parameters of distribution for each node: means $\mu_U, \mu_Y, \mu_{\mathbb{X}}$, variances σ_U^2, σ_Y^2 and covariance matrix $\sum_{\mathbb{X}}$. The covariance matrix $\sum_{\mathbb{X}}$ (3.14) is a symmetric matrix that represents variances of state variables and the power of their dependencies expressed by the covariances between all pairs of states.

During all experiments described in this thesis, the values of means were set to zero values. It is possible to provide identification procedure without this setting, but the resulting matrices will not correspond to the state matrices of a dynamic system. If the goal of a learning procedure is to find these matrices, then updating of means in learning procedure brings additional complications, since the state matrices have to be recalculated from the resulting weights. However, if a Bayesian network with learned parameters is used for further tasks (e.g. monitoring, control), then setting of means to zero values is not necessary. However, this setting brings additional unknown parameters that have to be estimated by the learning procedure and therefore more input data is required.

Since variances and covariance matrix are unknown in most practical applications, the most natural way to implement Bayesian network based system identification is to calculate them simultaneously with unknown parameters of a dynamic system. However, the implementation of this approach was connected with distinct numerical issues, in particular for considered systems of the third order the identification procedure failed in more than 80% of provided experiments. Even in cases, in which it succeeded to end properly, the precision of obtained estimates was low in many identification trials. This experiment is described in more details in the subsection 5.3.1.

We decided to consider variances and covariance matrix as tuning parameters in the effort to overcome the above mentioned issues. Even though their values are in most cases unknown, it is important to keep in mind that they represent the variance of noise in dynamic systems. This value can be roughly assessed from identification measurement. Fixing of the values of variances and covariance matrix during learning reduces the amount of parameters updated by the learning procedure that should have a positive effect on numerical properties.

To check this hypothesis, several learning settings, which will be referred to as scenarios, were designed:

- Scenario 1 - variances σ_U^2, σ_Y^2 are not fixed, covariance matrix $\Sigma_{\mathbb{X}}$ is not fixed.
- Scenario 2 – variances σ_U^2, σ_Y^2 are fixed to a certain value, covariance matrix $\Sigma_{\mathbb{X}}$ is assumed to be diagonal and the variances of individual state variables $\sigma_{x_1}^2, \sigma_{x_2}^2, \cdots, \sigma_{x_n}^2$ are fixed to a certain value.
- Scenario 3 – variances σ_U^2, σ_Y^2 are fixed to a certain value, covariance matrix $\Sigma_{\mathbb{X}}$ is not fixed.
- Scenario 4 - variances σ_U^2, σ_Y^2 are not fixed, covariance matrix $\Sigma_{\mathbb{X}}$ is assumed to be diagonal and the variances of individual state variables $\sigma_{x_1}^2, \sigma_{x_2}^2, \cdots, \sigma_{x_n}^2$ are fixed to a certain value.

- Scenario 5 - variances σ_U^2, σ_Y^2 are not fixed, covariance matrix $\Sigma_{\mathbb{X}}$ is assumed to be diagonal and the variances of individual state variables $\sigma_{x_1}^2, \sigma_{x_2}^2, \cdots, \sigma_{x_n}^2$ are not fixed.

- Scenario 6 - variances σ_U^2, σ_Y^2 are fixed to a certain value, covariance matrix $\Sigma_{\mathbb{X}}$ is assumed to be diagonal and the variances of individual state variables $\sigma_{x_1}^2, \sigma_{x_2}^2, \cdots, \sigma_{x_n}^2$ are not fixed.

The amount of unknown parameters updated by the learning procedure in each scenario for each considered type of a dynamic system can be found in the table 5.3. The character of behavior of a dynamic system (whether it is aperiodic or oscillate) does not influence the amount of unknown parameters, and therefore, they are unified to a single column in the table.

Table 5.3: Amount of unknown parameters in different learning scenarios

The type of a learning scenario	The type of a considered system			
	A system of the 1st order	A system of the 2nd order without direct feedthrough	A system of the 2nd order with direct feedthrough	A system of the 3rd order without direct feedthrough
Scenario 1 (without fixed parameters)	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, $C \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2, $\Sigma_x(\sigma_{x_1}^2)$ **Total: 6**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}$, σ_u^2, σ_y^2, $\Sigma_x(\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_1x_2})$ **Total: 13**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}, D \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2, $\Sigma_x(\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_1x_2})$ **Total: 14**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, $C \in \mathbb{R}^{1\times3}$, σ_u^2, σ_y^2, $\Sigma_x(\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_3}^2$, $\sigma_{x_1x_2}, \sigma_{x_1x_3}, \sigma_{x_2x_3})$ **Total: 23**
Scenario 2 (fixed σ_u^2, σ_y^2, Σ_x)	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, $C \in \mathbb{R}^{1\times1}$ **Total: 3**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}$ **Total: 8**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}, D \in \mathbb{R}^{1\times1}$ **Total: 9**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, $C \in \mathbb{R}^{1\times3}$ **Total: 15**
Scenario 3 (fixed σ_u^2, σ_y^2)	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, $C \in \mathbb{R}^{1\times1}$, $\Sigma_x(\sigma_{x_1}^2)$ **Total: 4**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}$, $\Sigma_x(\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_1x_2})$ **Total: 11**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}, D \in \mathbb{R}^{1\times1}$, $\Sigma_x(\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_1x_2})$ **Total: 12**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, $C \in \mathbb{R}^{1\times3}$, $\Sigma_x(\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_3}^2$, $\sigma_{x_1x_2}, \sigma_{x_1x_3}, \sigma_{x_2x_3})$ **Total: 21**
Scenario 4 (fixed Σ_x)	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, $C \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2 **Total: 5**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}$, σ_u^2, σ_y^2 **Total: 10**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}, D \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2 **Total: 11**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, $C \in \mathbb{R}^{1\times3}$, σ_u^2, σ_y^2 **Total: 17**
Scenario 5 (Σ_x is diagonal)	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, $C \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2$ **Total: 6**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}$, σ_u^2, σ_y^2, $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 12**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}, D \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 13**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, $C \in \mathbb{R}^{1\times3}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_3}^2$ **Total: 20**
Scenario 6 (fixed σ_u^2, σ_y^2, Σ_x is diagonal)	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, $C \in \mathbb{R}^{1\times1}$, $\sigma_{x_1}^2$ **Total: 4**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}$, $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 10**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}, D \in \mathbb{R}^{1\times1}$, $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 11**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, $C \in \mathbb{R}^{1\times3}$, $\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_3}^2$ **Total: 18**

Since experiments were provided on deterministic systems, the value of variance is, in fact, equal to zero. Since it is not possible in BNT to set the value of variance to zero, this parameter should be set to a small value. The range of values for all experiments was the following:

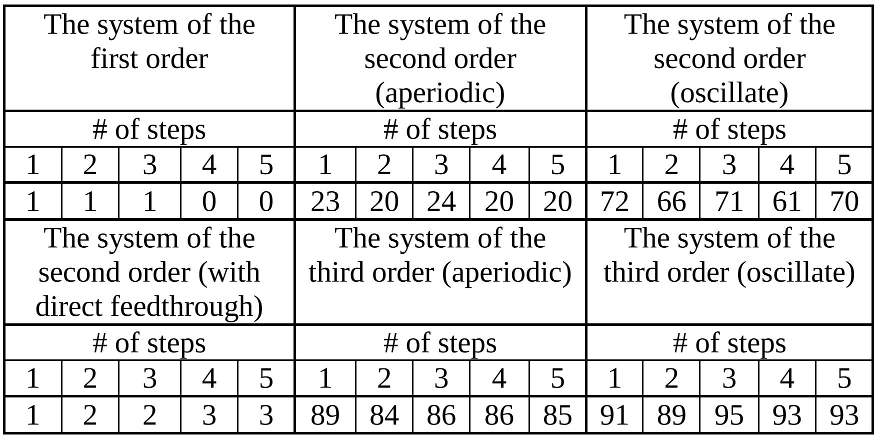
$$\sigma^2_{exp} = \begin{bmatrix} 0.1 & 0.01 & 0.001 & 0.0001 & 0.00001 \end{bmatrix}. \tag{5.3}$$

Experiments consisted of identification using datasets from 6 dynamic systems with 5 different input signals (refer to the subsection 5.2.2). Therefore, for learning scenarios 1 and 5 (where values of variances were not fixed) 30 bunches of identification experiments with 100 identification trials in a bunch were provided for each scenario. For learning scenarios 2, 3, 4 and 6 each bunch of identification experiments was provided for 5 different values of fixed variance σ^2_{exp}, therefore 150 bunches of experiments with 100 identification trials in a bunch were provided for each scenario.

5.3.1 Analysis of the learning scenario 1

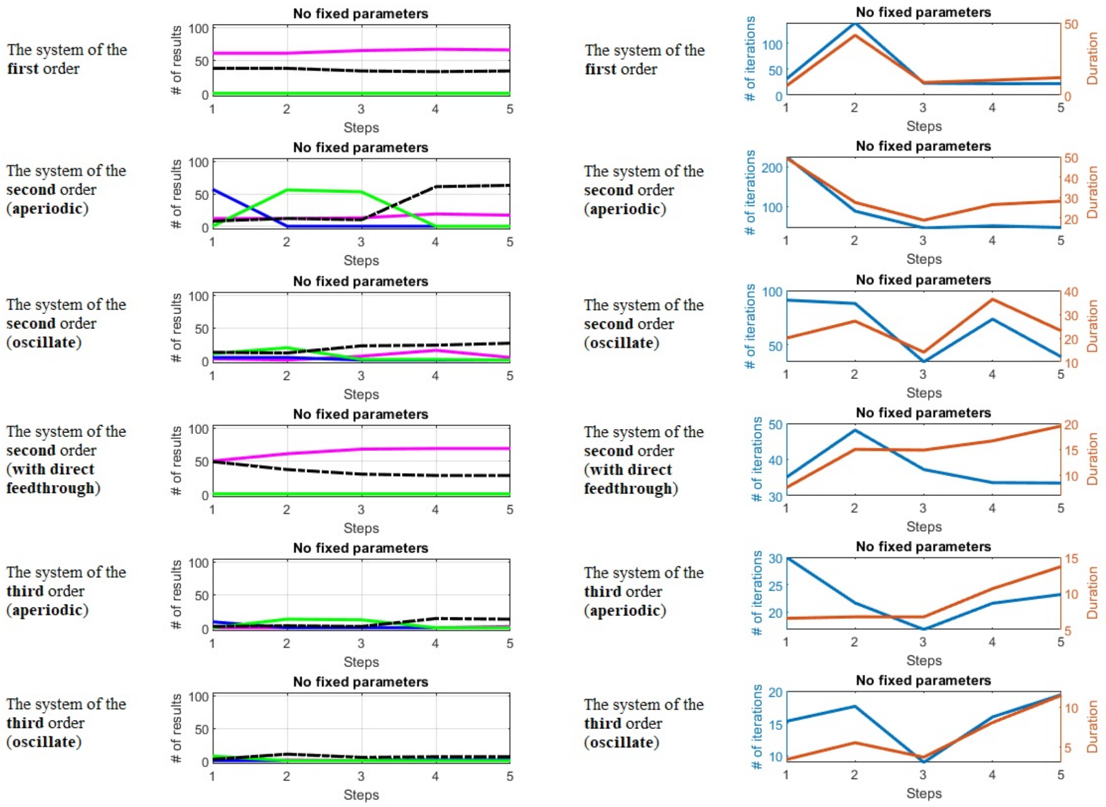
The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.8a. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.8b.

Table 5.4: Amount of fails during calculation in learning scenario 1



The system of the first order					The system of the second order (aperiodic)					The system of the second order (oscillate)				
# of steps					# of steps					# of steps				
1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	1	1	0	0	23	20	24	20	20	72	66	71	61	70
The system of the second order (with direct feedthrough)					The system of the third order (aperiodic)					The system of the third order (oscillate)				
# of steps					# of steps					# of steps				
1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	2	2	3	3	89	84	86	86	85	91	89	95	93	93

This learning scenario was the most numerically unstable. The amount of fails during execution of identification scripts is presented in the table 5.4. One of reasons



(a) The results of identification

(b) Numerical properties

Figure 5.8: Analysis of learning scenario 1

for such instability is the high amount of unknown parameters (state matrices, variances and covariance matrix). Therefore, it was unusable for systems of the third order. For the oscillate system of the second order more than a half of identification trials failed. In addition, from the trials which succeeded to finish properly, the majority did not cope to distinguish an oscillation from the influence of noise and hence resulted in aperiodic descriptions.

For the aperiodic system of the second order the percentage of non-fail results was considerably better. The majority of experiments had successful results, but for the '4 steps' and '5 steps' input signals there were many results that were classified as bad, but actual values of mean squared errors were slightly higher than the predefined bound for bad results $J = 0.05$ (refer to the figure 5.9 for the examples of such responses obtained for '4 steps' input signal). However, changing of this bound to higher value would bring additional complications to the evaluation of other scenarios, since for most of them this bound was optimal. Therefore, we kept the same value of this bound for all learning scenarios to make analysis consistent.

The scenario provided similar behavior for the aperiodic system of the third order

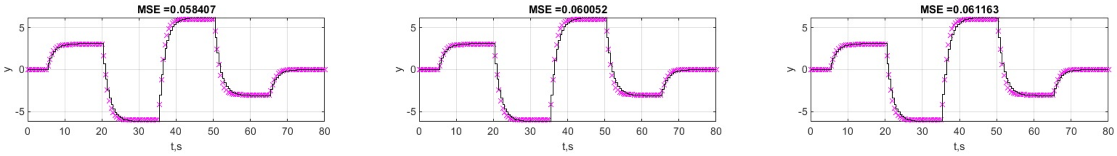


Figure 5.9: Examples of identification results classified to class 4

in successful identification trials. However, the amount of fails during identification make this scenario unusable for this type of dynamic systems.

The systems for which this scenario was relatively successful was the system of the first order and the system of the second order with direct feedthrough. For these systems the learning scenario succeeded to get more than half of perfect results of system identification.

The average duration of an identification trial was for most bunches equal to $10 - 20$ seconds. In some cases, it was slightly higher, up to 50 seconds.

5.3.2 Analysis of the learning scenario 2

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.10. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.11.

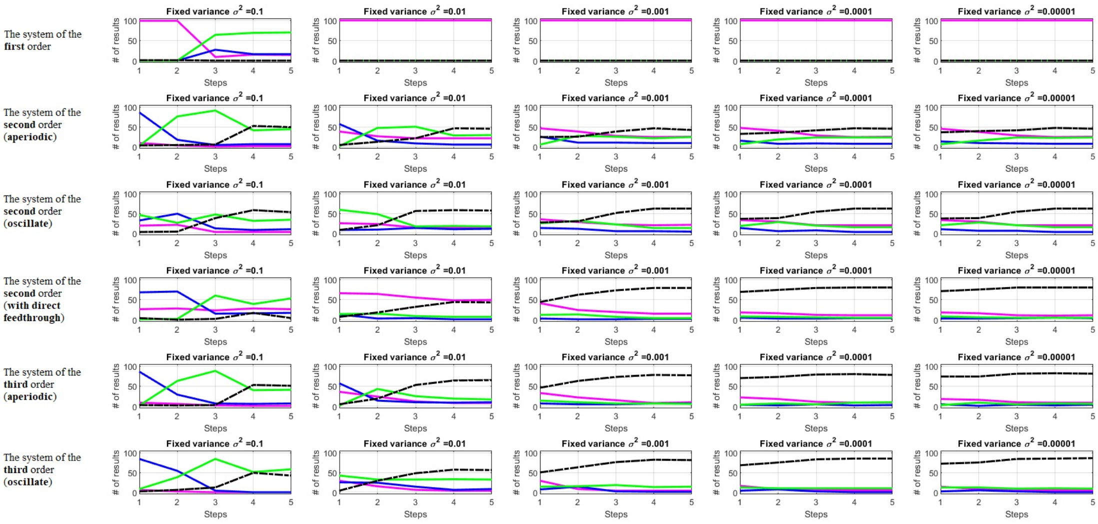


Figure 5.10: Results of identification provided using scenario 2

For the system of the first order the results were successful in the majority of cases. For the values of fixed variance $\sigma^2_{exp} < 0.1$ all results were perfect, for the value $\sigma^2_{exp} = 0.1$ the majority of results were successful, but the value of cost function increased with increasing amount of input data. The computation was stable, there were no fails and only 2 identification trials from 2500 were classified as bad. The average duration of an identification trial for $\sigma^2_{exp} < 0.1$ was only 1–2 seconds. Hence for the system of the first order this learning scenario can be recommended for usage with sufficiently low value of fixed variance.

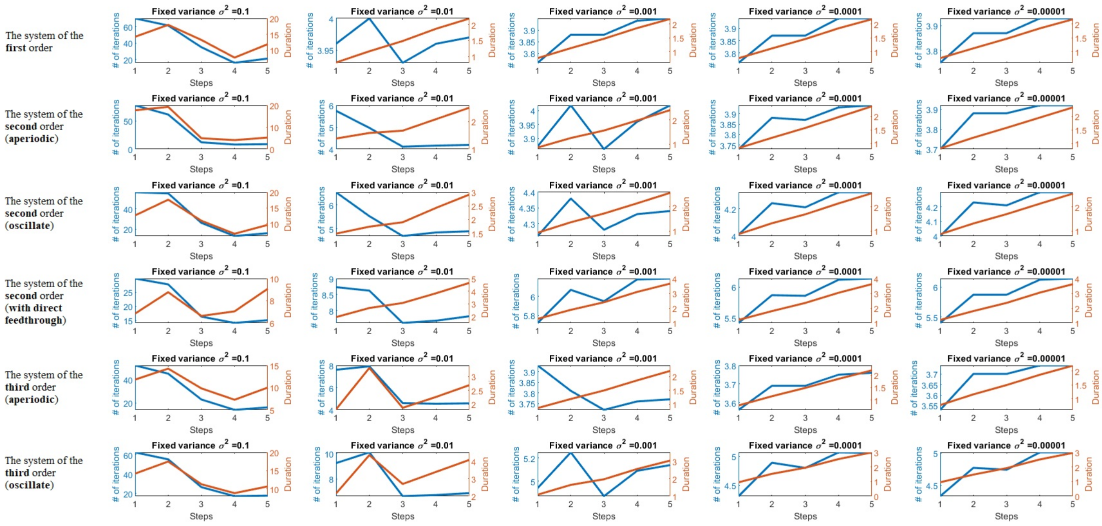


Figure 5.11: Numerical properties of scenario 2

For other systems this scenario was not as promising. For the aperiodic system of the second order the percentage of bad results (with high value of a cost function) did not exceed 50% for all bunches of identification trials. The amount of successful results decreased with increasing duration of simulated identification datasets and with decreasing value of fixed variance σ^2_{exp}.

For other dynamic systems the amount of unsuccessful results for $\sigma^2_{exp} < 0.01$ was higher than 50%, therefore we do not recommend the using of this scenario with such small value of fixed variance. It is also important to mention, that unsuccessful results are far from the original descriptions of dynamic systems. A lot of them are unstable and they sometimes lead to the oscillate description for aperiodic systems. The amount of unreasonable results increases with decreasing σ^2_{exp} and it does not get better with the increasing the amount of input data.

Further disadvantage of this scenario is related to its low ability to properly identify oscillate systems. It is not often seen from the value of the cost function, but visual comparison of input data and response of resulting system shows that the learning algorithm succeeded to include oscillate component into the description of identified dynamic systems in only half of cases. In other cases, it considered oscillations as the influence of noise.

The computation process in identification trials provided using scenario 2 was stable. Regardless numerical difficulties with calculations caused by appearance of singular matrices or matrices that are close to singular, the program succeeded to provide 15 000 identification trials with only one fail (during identification of the system of the second order with direct feedthrough for '1 step' input signal and $\sigma^2_{exp} = 0.1$).

The average duration of an identification trial did not exceed 5 seconds for settings $\sigma^2_{exp} < 0.1$. For the setting $\sigma^2_{exp} = 0.1$ this value was a little bit higher, up to 20 seconds.

5.3.3 Analysis of the learning scenario 3

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.12. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.13.

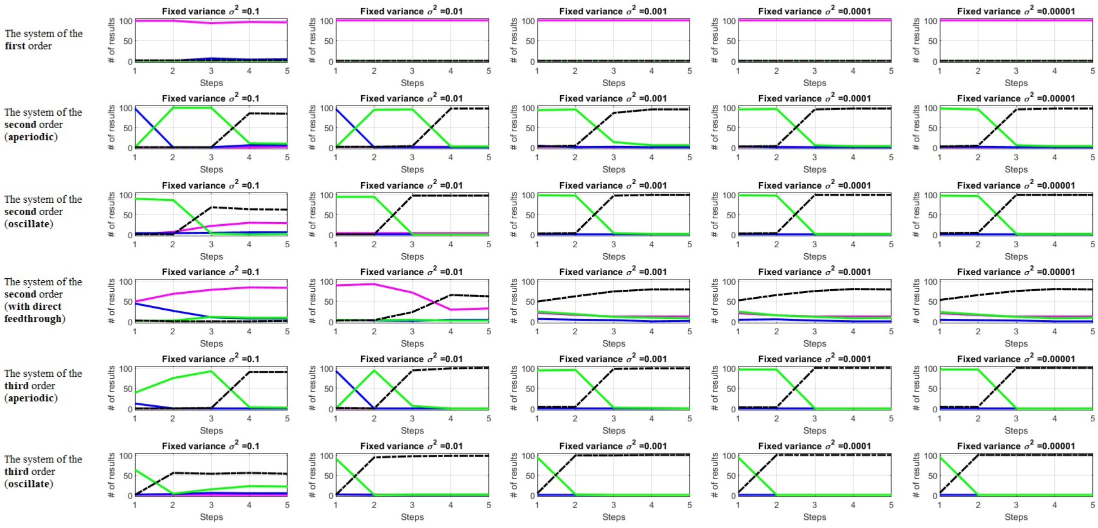


Figure 5.12: Results of identification provided using scenario 3

For the system of the first order this learning scenario succeeded to obtain perfect results for the value of fixed variance $\sigma^2_{exp} < 0.1$. For $\sigma^2_{exp} = 0.1$ there were several bad results (one in each bunch) and several descriptions which were classified as good. Therefore, for the system of the first order this scenario can be recommended.

Big ratio of successful results was also obtained for the system with direct feedthrough in setting $\sigma^2_{exp} = 0.1$, the majority of results were classified as perfect or good.

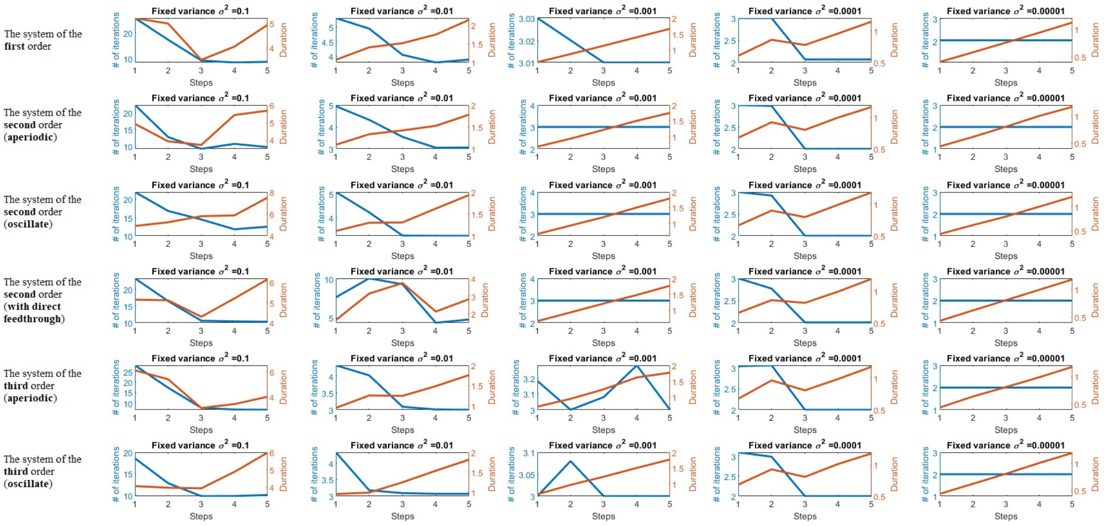


Figure 5.13: Numerical properties of scenario 3

For other systems the results were ambiguous, in particular for aperiodic systems of the second and the third order. Analogously to the previous scenario, the significant amount of bad results was obtained for $\sigma^2_{exp} < 0.1$. But unlike scenario 2, in which the estimates that were classified as bad corresponded to unreasonable responses distant from the original datasets, in this scenario the bad results often corresponded to the appropriate behavior of considered dynamic systems, but the value of a cost function was slightly higher than pre-defined bound 0.05. (refer to the figure 5.9 for the examples of such responses). There were not more than several truly inappropriate results per a bunch of identification trials. Unfortunately, this was not the case for a dynamic system with direct feedthrough, where a lot of unreasonable results appeared. Therefore, this scenario can be recommended for aperiodic systems without direct feedthrough with relatively low demands on the precision of obtained results.

For oscillate systems this scenario cannot be recommended, since it failed to catch the oscillate nature of identified systems in the majority of cases. These systems were in most cases identified by the state space that corresponded to a dynamic system with aperiodic behavior. Therefore, even the results classified as successful often correspond to the inappropriate description of a considered dynamic system.

Another issue of this scenario is numerical stability. The amount of fails in each bunch of identification trials is presented in the table 5.5.

Table 5.5: Amount of fails during calculation in learning scenario 3

	The system of the first order					The system of the second order (aperiodic)					The system of the second order (oscillate)				
	# of steps					# of steps					# of steps				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
$\sigma^2 = 0.1$	0	0	0	0	0	3	1	1	0	1	8	5	5	3	4
$\sigma^2 = 0.01$	0	0	0	0	0	2	4	1	0	0	3	3	0	0	0
$\sigma^2 = 0.001$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$\sigma^2 = 0.0001$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$\sigma^2 = 0.00001$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	The system of the second order (with direct feedthrough)					The system of the third order (aperiodic)					The system of the third order (oscillate)				
	# of steps					# of steps					# of steps				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
$\sigma^2 = 0.1$	4	4	1	0	0	49	25	7	7	8	36	40	28	19	22
$\sigma^2 = 0.01$	1	0	1	0	0	6	6	0	1	0	7	5	2	1	1
$\sigma^2 = 0.001$	0	0	0	0	0	1	1	0	0	1	2	0	1	0	0
$\sigma^2 = 0.0001$	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
$\sigma^2 = 0.00001$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

In particular, for setting $\sigma^2_{exp} = 0.1$ the algorithm failed to converge for dynamic systems of the third order in many cases. However, the increasing of input dataset had positive effect on numerical stability. Decreasing of the value of fixed variance σ^2_{exp} also decreased the amount of fails up to zero. Hence, it can be concluded that in this scenario there is a trade-off between numerical stability and the precision of obtained estimates.

The advantage of this scenario is its speed. In most cases, the average duration of an identification trial was only several seconds.

5.3.4 Analysis of the learning scenario 4

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.14. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.15.

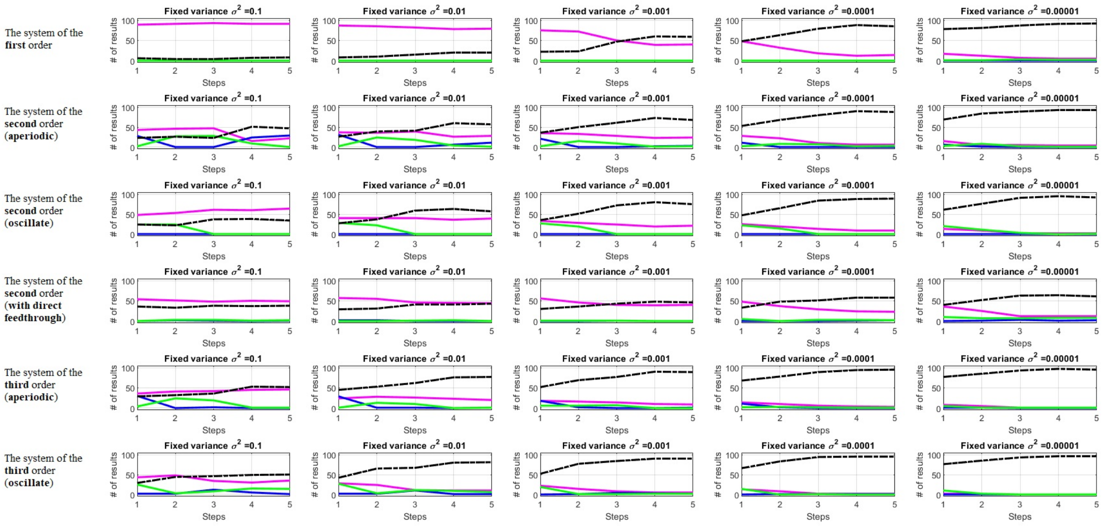


Figure 5.14: Results of identification provided using scenario 4

The implementation of this scenario was accompanied with the high amount of numerical problems caused by the singularity of matrices used for estimation of parameters. In some cases, the algorithm still managed to converge, but for some systems (e.g. the system of the second order with direct feedthrough) these problems persisted even with increasing of input dataset. The information about fails in each bunch of identification trials is presented in the table 5.6.

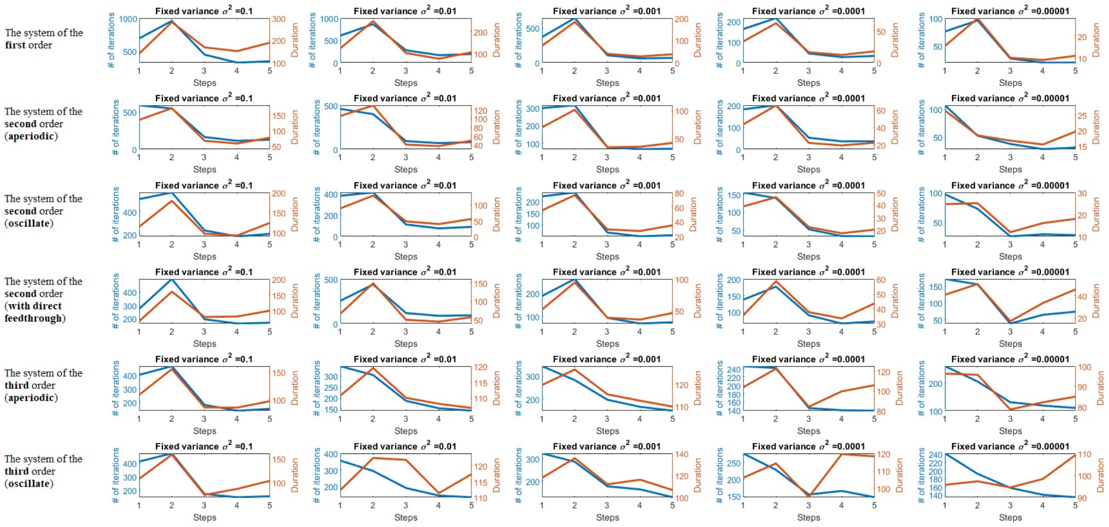


Figure 5.15: Numerical properties of scenario 4

However, the precision of obtained results was in average better than in the previous scenarios. For the value of fixed variance $\sigma^2_{exp} = 0.1$ the algorithm succeeded to obtain more than 50% of successful results, from which the majority had low value of mean squared error and hence is considered perfect. The amount of unsuccessful results increased with increasing amount of input data and decreasing value of fixed variance σ^2_{exp}. In this scenario, similarly to scenario 2, decreasing of the value of fixed variance lead to the increasing amount of inappropriate responses of systems with identified parameters (e.g. oscillate responses for aperiodic systems or unstable responses).

Table 5.6: Amount of fails during calculation in learning scenario 4

	The system of the first order					The system of the second order (aperiodic)					The system of the second order (oscillate)				
	# of steps					# of steps					# of steps				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
$\sigma^2 = 0.1$	5	5	3	2	1	4	1	2	1	2	4	1	2	2	2
$\sigma^2 = 0.01$	5	5	3	2	1	4	1	2	4	3	5	1	1	1	4
$\sigma^2 = 0.001$	3	5	3	1	1	6	2	2	1	3	5	2	4	1	4
$\sigma^2 = 0.0001$	4	5	3	0	1	6	2	3	2	3	6	2	3	3	2
$\sigma^2 = 0.00001$	4	6	3	2	1	7	2	4	2	2	6	4	4	3	5
	The system of the second order (with direct feedthrough)					The system of the third order (aperiodic)					The system of the third order (oscillate)				
	# of steps					# of steps					# of steps				
	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
$\sigma^2 = 0.1$	10	11	10	12	10	1	3	1	1	1	1	3	0	1	0
$\sigma^2 = 0.01$	12	12	12	12	13	1	5	1	1	1	1	6	1	0	2
$\sigma^2 = 0.001$	13	17	15	13	14	6	7	2	0	1	7	7	3	0	1
$\sigma^2 = 0.0001$	14	15	17	14	15	5	7	3	1	1	7	8	2	2	2
$\sigma^2 = 0.00001$	14	15	15	15	17	9	8	5	2	4	12	11	7	4	4

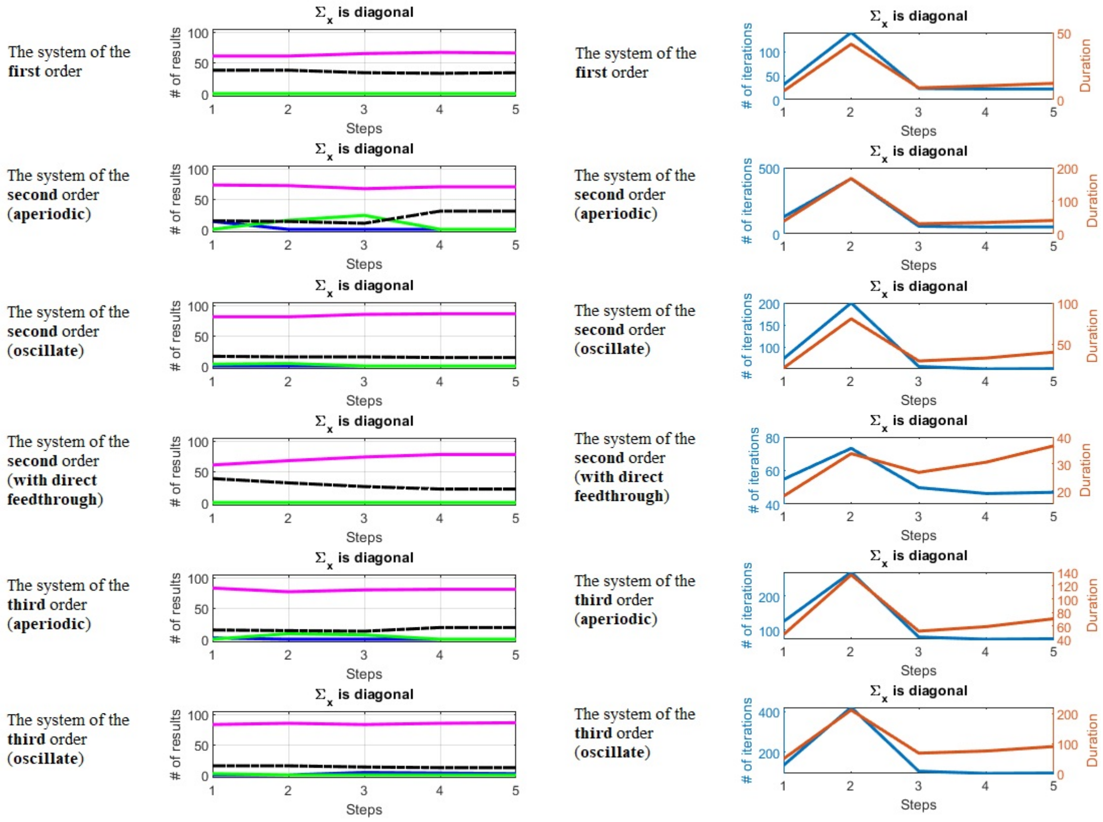
The system, for which this scenario provided distinctively worse estimates was the system of the first order, in particular for higher amounts of input steps and lower values of fixed variance.

During identification of oscillate systems, the learning scenario managed to catch their oscillate behavior in more than the half of successful identification trials, which is considerably better result than during the application of scenario 3.

The average duration of an identification trial was several minutes in almost each bunch of experiments. It was particularly long for responses on '2 steps' input signal.

5.3.5 Analysis of the learning scenario 5

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.16a. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.16b.



(a) The results of identification

(b) Numerical properties

Figure 5.16: Analysis of learning scenario 5

The settings of this learning scenario are similar to the settings of scenario 1 (variances σ_U^2, σ_Y^2 are not fixed, the covariance matrix $\Sigma_{\mathbb{X}}$ is not fixed) with the only difference in the type of the covariance matrix. In this scenario it is assumed to be diagonal that corresponds to the case of independent state variables. Consequently, the algorithm has less values to update at each iteration since it does not have to calculate covariances between state variables.

For the system of the first order, the identification procedure was analogous to the one used in scenario 1, since the covariance matrix for the system of the first order is represented by the variance of a single state variable.

This scenario provided the most successful results among all scenarios. The amount of perfect results in different bunches was impressive in comparison with other learning scenarios. For example, for dynamic systems of the third order perfect results were obtained in more than 80% of cases. For other systems this value was slightly worse, but it was higher than 60% in all bunches of identification trials. The increasing of the length of input dataset slightly improved the amount of successful results for the majority of considered dynamic systems.

Identification of oscillate dynamic systems was provided without noticeable complications. In some cases, the learning scenario approximated the data with aperiodic response, but this situation was rather rare.

For the majority of considered dynamic systems this learning scenario provided the best results of identification. But even for those systems for which it did not (the system of the first order and the system of the second order with direct feedthrough) the amount of perfect results was relatively high in all identification bunches. Therefore, this learning scenario is recommended for any type of identified dynamic systems.

The shortcoming of this learning scenario is its duration. In most cases, it took EM algorithm 1 up to 3 minutes to converge. This duration was lower for the system of the first order and the system of the second order with direct feedthrough (up to 45 seconds), but at the same time, the worse ratio of perfect results was obtained for these systems.

The learning algorithm had stable behavior. Only 3 identification trials ended up with fails among 3000 trials provided using this scenario. They all appeared during identification of the system of the first order (1 for '1 step' input signal, 1 for '2 steps' input signal and 1 '3 steps' input signal).

This scenario was chosen for system identification in all further experiments since it provided the best estimates of unknown parameters among all learning scenarios.

5.3.6 Analysis of the learning scenario 6

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.17. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.18.

The settings of this learning scenario are similar to the settings of scenario 3 (variances σ_U^2, σ_Y^2 are fixed, the covariance matrix $\Sigma_{\mathbb{X}}$ is not fixed) with the only difference in the type of the covariance matrix. In this scenario it is assumed to be diagonal that corresponds to the case of independent state variables. Consequently, the algorithm has less values to update at each iteration since it does not have to

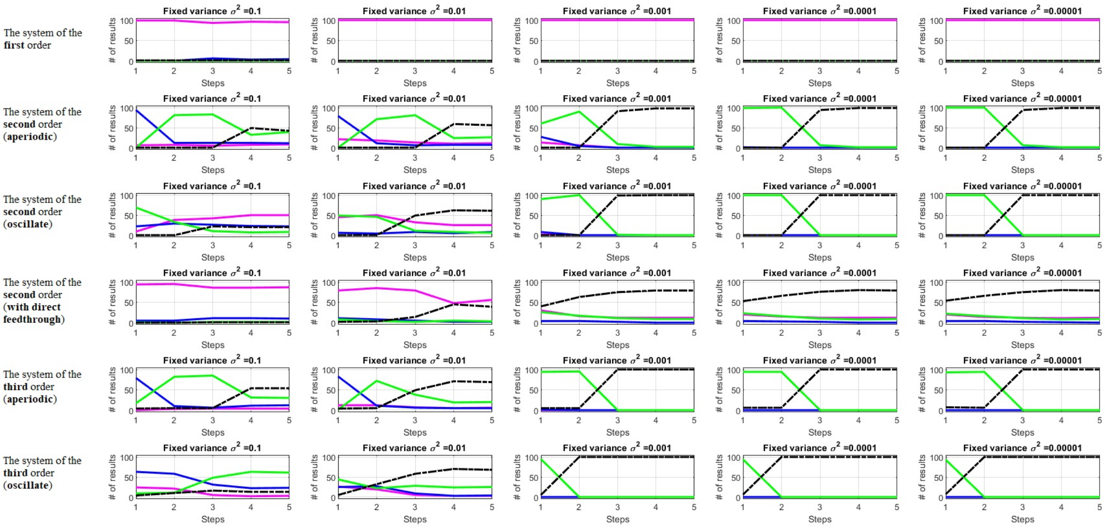


Figure 5.17: Results of identification provided using scenario 6

calculate covariances between state variables.

For the system of the first order the identification procedure was analogous to the one used in scenario 3, since the covariance matrix for the system of the first order is represented by the variance of a single state variable.

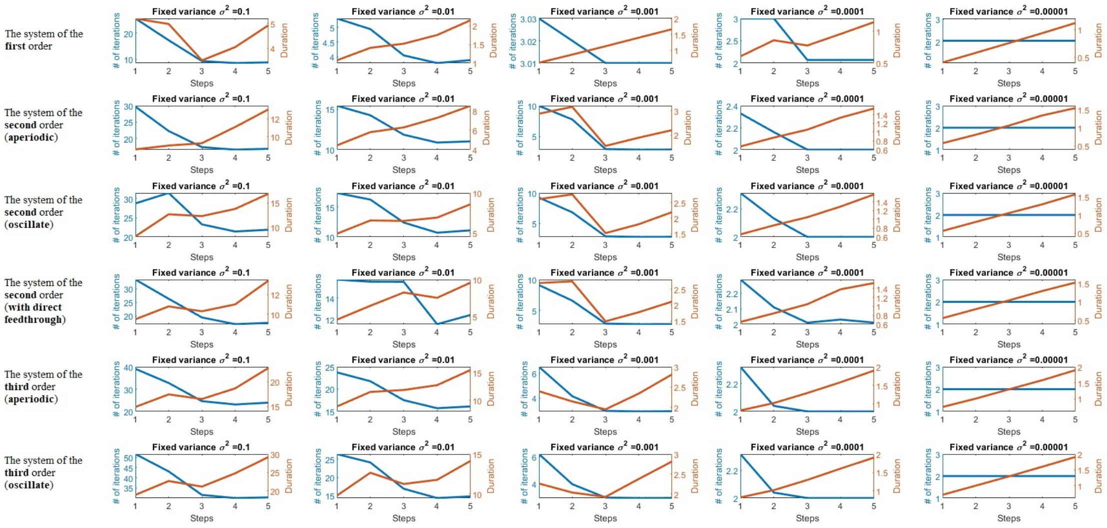


Figure 5.18: Numerical properties of scenario 6

This algorithm was considerably more stable, there was not a single fail among all 15 000 identification trials. The implementation of this scenario also resulted in better estimates of parameters. Analogously to the scenario 2, many results that were classified as bad have reasonable responses, but slightly higher value of the cost function. In this scenario, the ratio of reasonable results was higher than in scenario 3.

Similarly to scenario 3, this learning scenario had issues during identification of oscillate dynamic systems, but for this scenario the ratio of successful results was considerably higher. In particular, it was the highest for the setting $\sigma_{exp}^2 = 0.1$ and it increased with increasing length of a dataset. Therefore, unlike the scenario 3, this scenario can be used for oscillate dynamic systems. To increase chances to obtain proper estimates of parameters, we recommend not to use too small value of fixed variance σ_{exp}^2.

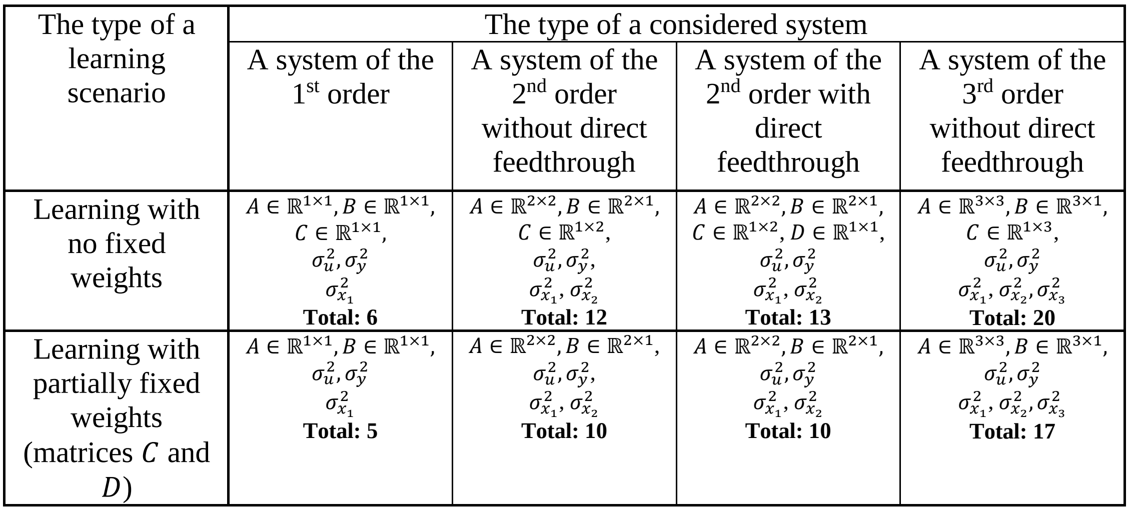
The average duration of an identification trial was in most cases only several seconds. For the setting $\sigma_{exp}^2 = 0.1$ it was slightly higher (up to 30 seconds), but at the same time, this setting ended up with the most precise estimates.

5.4 Influence of reducing the amount of unknown parameters

In the previous section, we experimented with the reducing of the amount of unknown parameters by the setting of variances and the covariance matrix to different values. The learning scenario that was chosen based on provided experiments was the one in which the covariance matrix is set to have a diagonal form and all variances (for state variables, input and output signals) are not fixed and hence updated simultaneously with unknown parameters of a dynamic system (refer to the section 5.3.5). Reducing of the amount of unknown parameters can be also achieved by the setting of weights for some nodes in a corresponding Bayesian network. This setting can be used, if the prior knowledge about a dynamic system is known.

The best way how to reduce the amount of unknown parameters is to use canonical forms of the state space representation. The resulting state matrices provide unequivocal description of a dynamic systems and they explicitly contain the coefficients of the corresponding difference equation. Unfortunately, it was not possible to explore this setting using BNT, since it is not possible to fix separate weights using this tool. However, it is possible to fix all weights for a node. Therefore, the influence of the reducing the amount of unknown parameters was studied for a Bayesian network with fixed weights of the node that corresponds to the output signal, i.e. the matrix $\mathbb{C}$ and the matrix $\mathbb{D}$ for a system with direct feedthrough (for other systems, its value is automatically fixed to zero). The resulting amount of unknown parameters after reduction can be found in the table 5.7.

Table 5.7: The amount of unknown parameters in different settings



The type of a learning scenario	The type of a considered system			
	A system of the 1st order	A system of the 2nd order without direct feedthrough	A system of the 2nd order with direct feedthrough	A system of the 3rd order without direct feedthrough
Learning with no fixed weights	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, $C \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2$ **Total: 6**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}$, σ_u^2, σ_y^2, $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 12**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, $C \in \mathbb{R}^{1\times2}, D \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 13**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, $C \in \mathbb{R}^{1\times3}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_3}^2$ **Total: 20**
Learning with partially fixed weights (matrices C and D)	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2$ **Total: 5**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, σ_u^2, σ_y^2, $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 10**	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2, \sigma_{x_2}^2$ **Total: 10**	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}$, σ_u^2, σ_y^2 $\sigma_{x_1}^2, \sigma_{x_2}^2, \sigma_{x_3}^2$ **Total: 17**

We used two settings for fixed weights. In the first setting, it was assumed that the values of parameters in matrices are known from expert knowledge regarding a considered dynamic system. In the second setting, we assumed that there is no prior information regarding the parameters and therefore, their values are sampled from the standard normal distribution and fixed to this sampled value. Consequently, these values are not updated by the learning procedure.

Additional experiments consisted of identification using datasets from 6 dynamic systems with 5 different input signals (refer to the subsection 5.2.2) for each setting. Therefore, 60 bunches of identification experiments with 100 identification trials in a bunch were provided for this experiment. The comparison of results with a setting with no fixed parameters (obtained in experiment described in the subsection 5.3.5) are presented on the figure 5.19 and the figure 5.20. The values of matrices $\mathbb{C}$ and $\mathbb{D}$ are shown in the titles above plots on the figures.

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.19. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.20.

For the setting with assumed presence of expert knowledge, we succeeded to get undoubtable increasing in the efficiency of system identification for two out of six considered systems: the system of the first order and the system of the second order with direct feedthrough. For the system of the first order this setting provided perfect estimates of parameters in all identification trials. The behavior of the

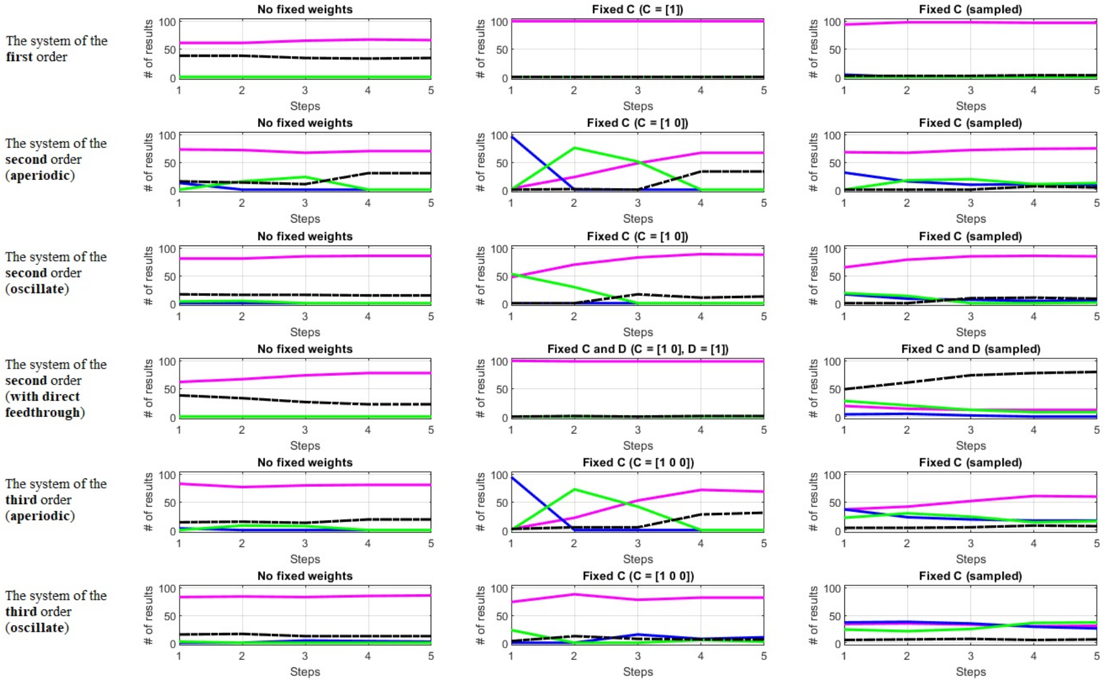


Figure 5.19: Results of identification with reduced amount of parameters

system of the second order with feedthrough was perfectly identified in 99% of cases. For other systems the influence is not as unambiguous. The precision of obtained estimates evidentially decreased, however the amount of bad results also decreased in the majority of bunches of identification experiments. In addition, this setting considerably decreased the amount of unreasonable results (e.g. non-stable results, zero-valued parameters) for all dynamic systems. Hence, the majority of identification trials classified as bad had slightly higher value of the associated mean squared error.

Another advantage, which followed from the reducing of the amount of unknown parameters, was the decreasing of the duration of identification trial for all considered systems. In many bunches, identification trials were several times faster with reduced amount of parameters (up to 5 times) than identification with no fixed weights.

The performance of the setting with fixing the parameters to values sampled from the standard normal distribution was similar to the previous setting. The precision of obtained results decreased, but the ratio of bad results and unreasonable responses of resulting models also decreased. Their ratio was approximately the same, as in the setting with included expert knowledge. The only system, for which these statements were not true, was the system of the second order with direct feedthrough. In these bunches of identification trials, the amount of unreasonable responses considerably increased comparing with the setting without fixed parameters, refer to the figure 5.21 for the examples of such responses.

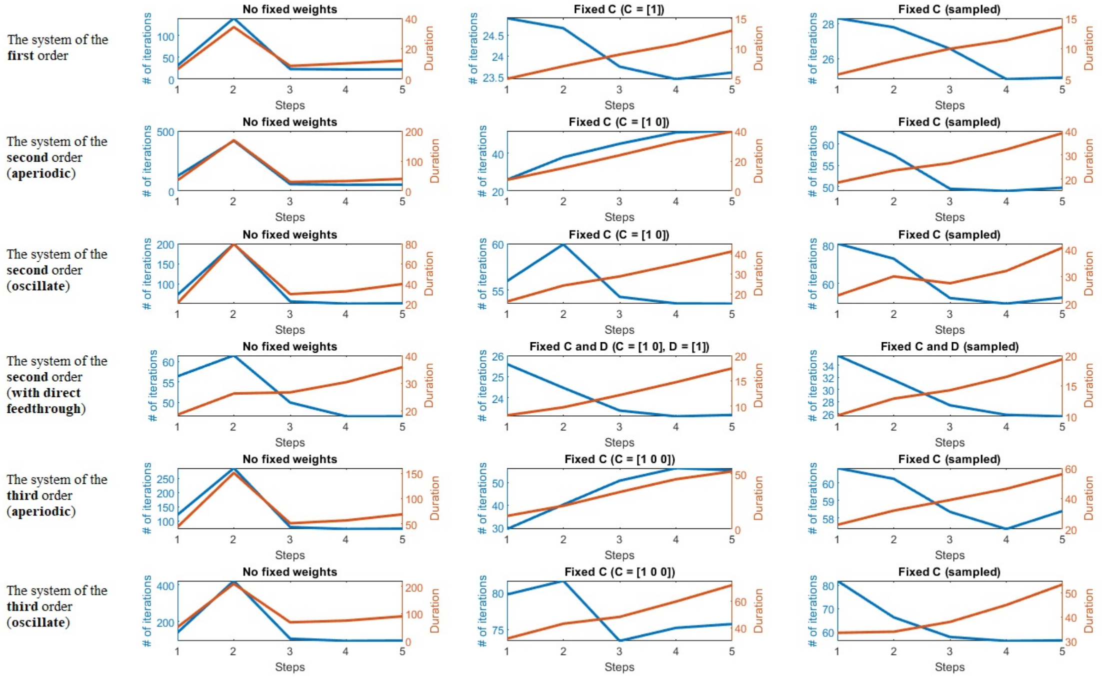


Figure 5.20: Numerical properties of identification with reduced amount of parameters

The numerical properties of two settings with reduced number of parameters were comparable. The average duration of an identification trial and the average amount of required iterations were in the same range.

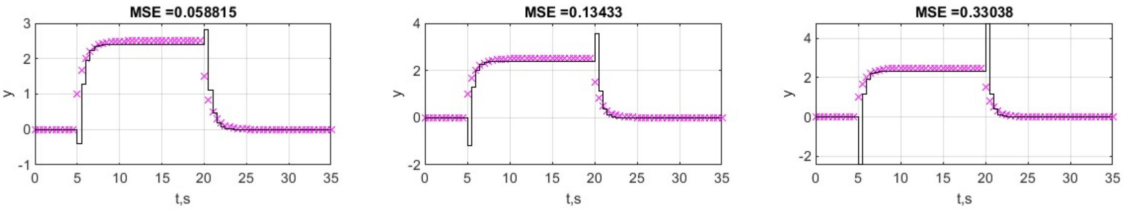


Figure 5.21: Examples of unreasonable results for the system with direct feedthrough

5.5 Influence of initial parameters on the results of identification

For the cases, in which a Bayesian network does not have hidden nodes and the dataset used for learning is complete, the maximum likelihood estimation of parameters is used for learning. In this setting, the estimates converge to the global optimum regardless the initial values of unknown parameters. This is applicable, for example, a Bayesian network based difference equation model.

For state space model, however, the unknown values of state variables have to be calculated simultaneously with the parameters of a considered dynamic system and therefore, the EM algorithm has to be used. It looks for the maximum of log likelihood function iteratively. Since the log likelihood function tends to have local optima, the choice of initial values of parameters is a key challenge in the learning procedure. If the initial values are close to true values of parameters, then the EM algorithm will converge fast to the global optimum. In other cases, however, reaching of the global optimum cannot be guaranteed.

Zero values of initial parameters are not the appropriate choice, since for most types of dynamic systems they provided zero estimates after several iterations, refer to the figure 5.22 for results from such setting. Only for the system of the second order with direct feedthrough the resulting estimates were not equal to zero, but they were far from the parameters of a considered system.

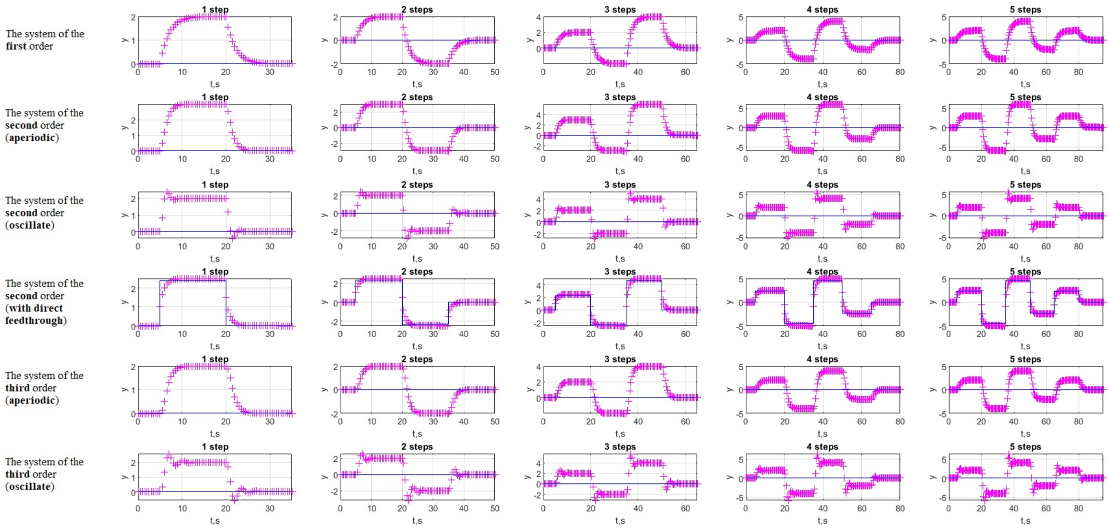


Figure 5.22: Results of identification for zero initial values of parameters

By default, the initial values of parameters in BNT are sampled from the standard normal distribution. The value from a standard normal distribution is assigned not only to the weights of a node, but to means, variances and covariance matrix, if they are not specified explicitly during description of a node's CPD. The goal of experiments described in this section was to explore, if we can obtain better percentage of successful results with initial values of parameters sampled from different type of probability distribution. In the first experiment, initial values were sampled from a normal distribution with different values of variances: $\sigma^2 = 0.5, 1, 2$. For each value of variance identification experiments were provided for 6 considered dynamic systems and 5 considered input signals. In total, 90 bunches of identification trials were provided, each bunch contained 100 trials.

In the second experiment, the type of used distribution was changed. Initial parameters were sampled from a uniform distribution with comparable range. The histograms of the values obtained from a random generator of normally distributed values are presented on the figure 5.23 (100 values in each case).

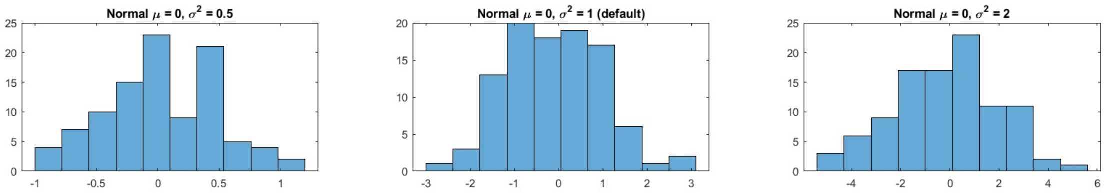


Figure 5.23: Values from random generator with different variances

The intervals of uniform distribution, which were chosen for further experiments, were: $< -1; 1 >$, $< -2; 2 >$ and $< -4; 4 >$. For these experiments, 90 bunches of identification trials were provided, each bunch contained 100 trials. The histograms of the values obtained from a random generator of uniformly distributed values are presented on the figure 5.24 (100 values in each case).

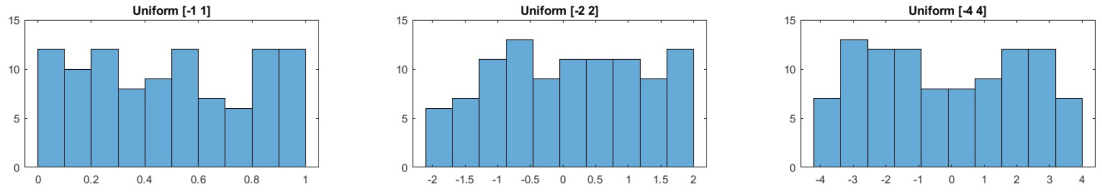


Figure 5.24: Values from random generator with different ranges

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.25. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.26.

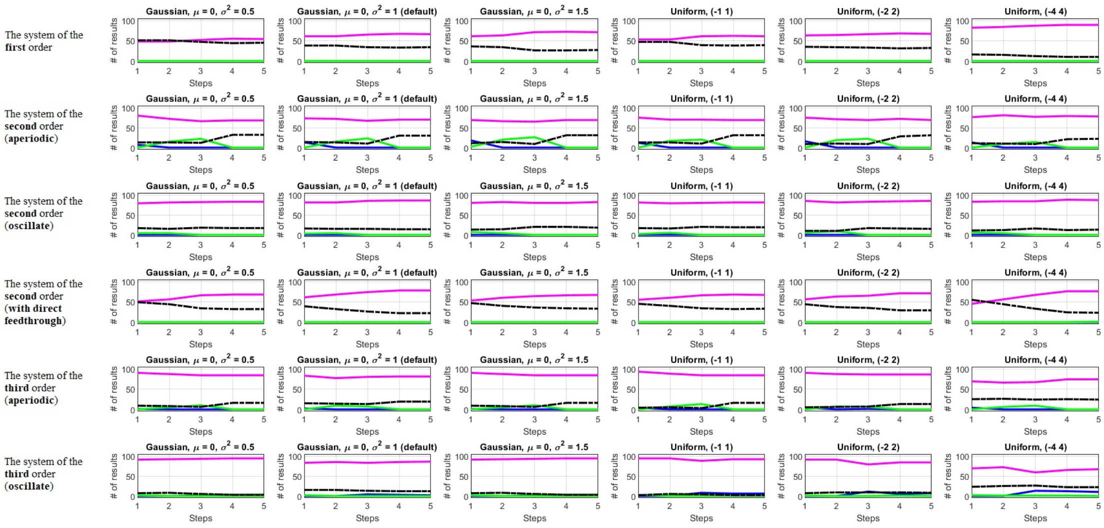


Figure 5.25: Results of identification for different types of distribution from which initial parameters were sampled

The results of identification for the oscillate system of the second order did not differ remarkably for different types of probability distributions and for different ranges. For other dynamic systems the percentage of perfect results slightly differed in different settings, but these results cannot be generalized over all considered types of dynamic systems, since each of them had best results in different settings.

In most cases, increasing of the range of variables caused notable increasing both in average iteration amount and in the average duration of the identification trial. Changing of the type of probability distribution used for sampling of initial values of parameters did not have remarkable effect on numerical properties of the learning algorithm.

Taking into account the impossibility of generalization, the non-remarkable changes in precision and the increase in both the amount of iterations and the average duration, we can conclude, that the changing of the type of used distribution and the range do not improve the learning procedure used for system identification.

The aim of the next experiment was to explore the influence of the mean of probability distribution used for sampling. Since there was almost no difference in the ratio of successful results between different types of considered distributions (nor-

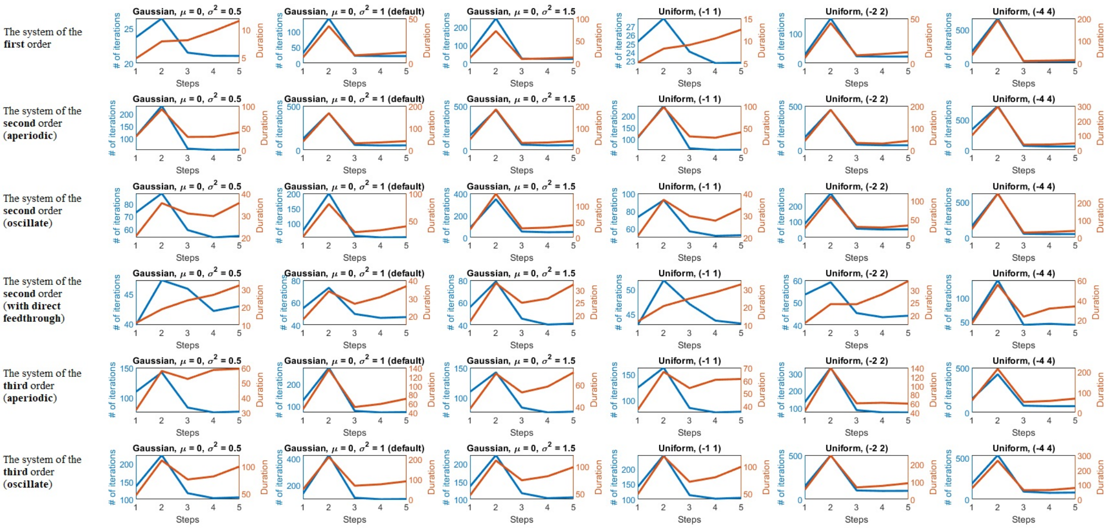


Figure 5.26: Numerical properties of identification with different types of distribution from which initial parameters were sampled

mal or uniform), in this experiment only normal distribution with different values of mean ($\mu = -2, -1, 0, 1, 2$) was used for sampling the initial values of parameters. For this experiment, 150 bunches of identification trials were provided, each bunch contained 100 trials. The histograms of the values obtained from a random generator of normally distributed values with different means are presented on the figure 5.27 (100 values in each case).

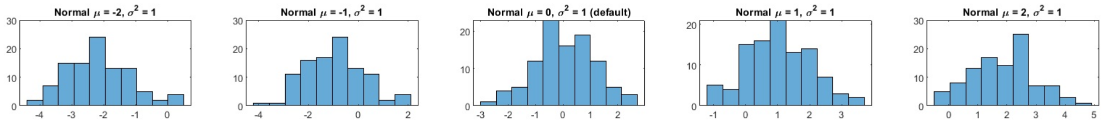


Figure 5.27: Values from random generator with different means

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.28. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch is presented on the figure 5.29.

For the system of the first order, the change of mean for distribution used for sampling of initial values of parameters remarkably improved the amount of perfect results. In particular, for settings $\mu = 1$ and $\mu = 2$ the ratio of perfect results was higher than 90% for each identification bunch, whereas with the default setting it was only slightly higher than 60%. Increasing of mean had also positive effect on

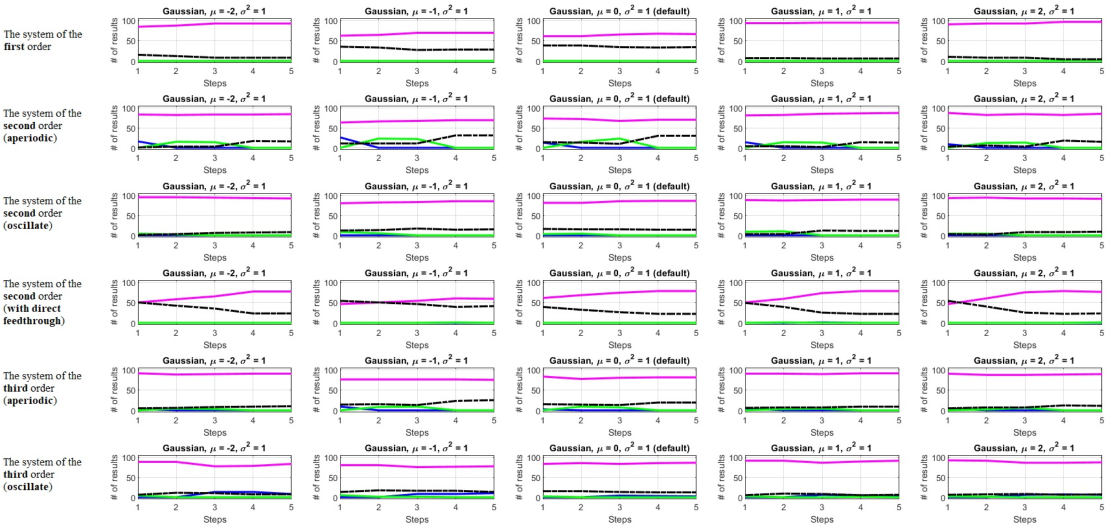


Figure 5.28: Results of identification for different means of distribution from which initial parameters were sampled

the amount of perfect results in all dynamic systems without direct feedthrough. For a system with direct feedthrough the default setting provided the most precise results of identification.

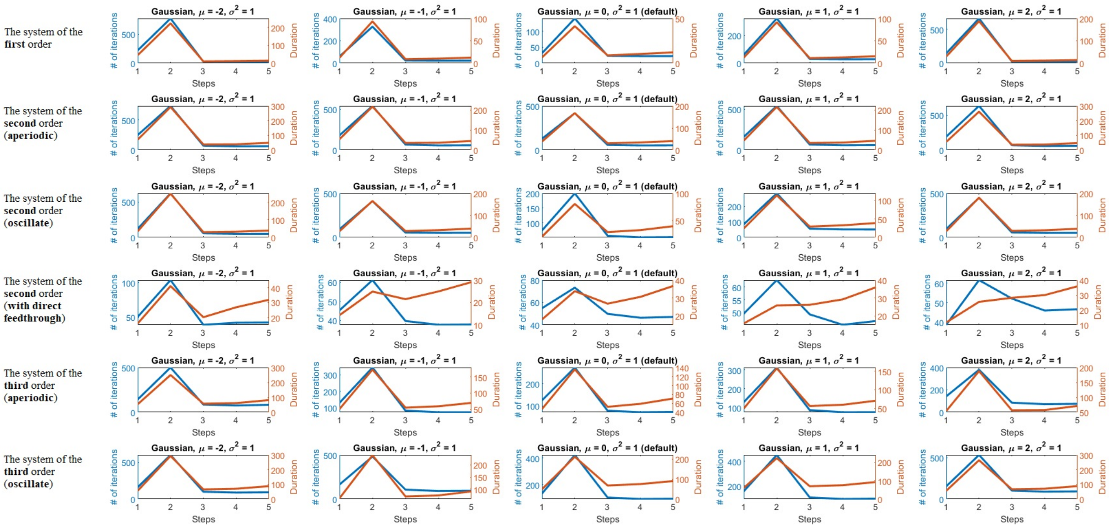


Figure 5.29: Numerical properties of identification with different means of distribution from which initial parameters were sampled

However, the obtained results cannot be generalized over all dynamic systems, since only one representative from each type was considered in this thesis. But this experiment proves that changing of the mean of distribution from which initial parameters are sampled is a perspective step to increase the ratio of successful identification results.

The disadvantage of non-default settings for a mean is worse numerical properties. For all dynamic systems without direct feedthrough moving of mean away from zero caused notable increasing in both the average amount of iterations and the average duration of an identification trial.

5.6 Identification of stochastic systems

Unlike theoretical responses, practical measurements provide noisy signals. Therefore, it is crucial to explore the efficiency of Bayesian networks in the identification of stochastic systems for their implementation in practical applications.

For the purposes of this experiment, the responses of the considered dynamic systems was enriched by noise with the different value of variance: $\sigma_\epsilon^2 = 0, 0.01, 0.02, 0.05, 0.1$. The noise was generated from a generator of normally distributed values with zero value of mean and the corresponding value of variance. The noise was added both to the output signal and to each state variable. Resulting datasets were used for identification of 6 considered systems (table 5.1) with 5 types of input signal (table 5.2). In total, 150 bunches of identification trials were provided in this experiment, with 100 identification trials in a bunch.

The amounts of results classified to 4 different classes based on the mean squared error (refer to the subsection 5.2.3) are presented on the figure 5.30.

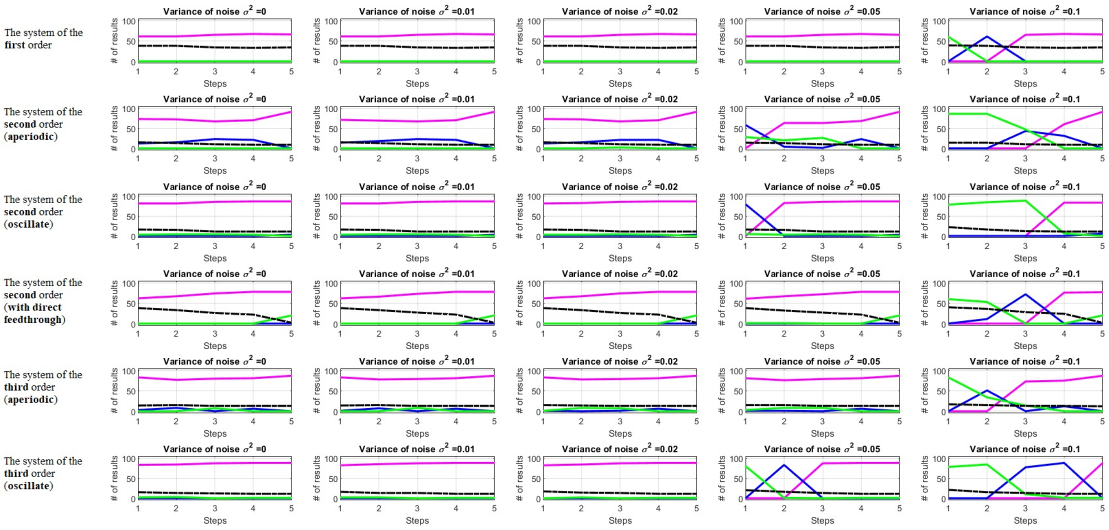


Figure 5.30: Results of identification for stochastic systems

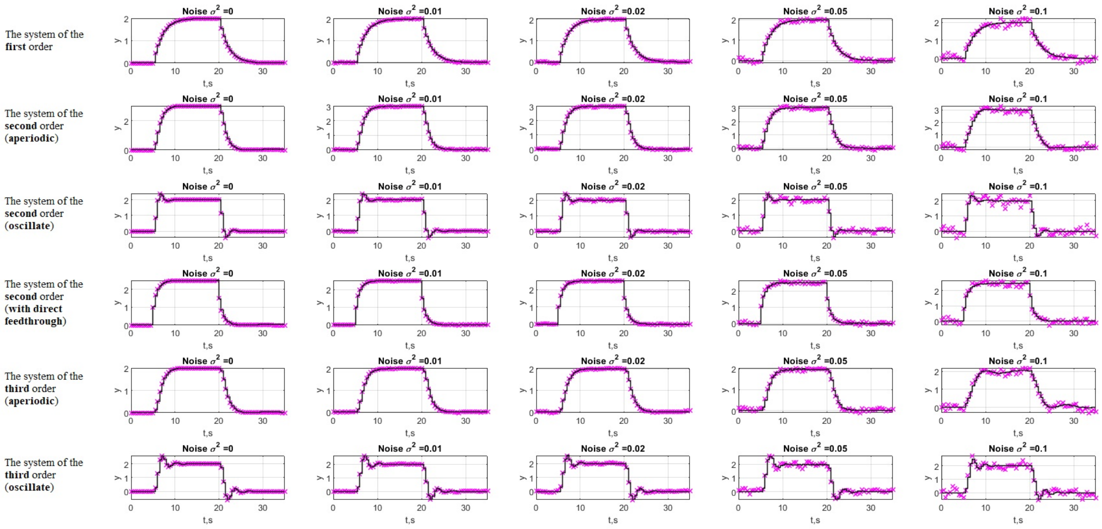


Figure 5.31: Results of BNSI for '1 step' input signal

For stochastic systems a trial was considered perfect if the mean squared error was smaller than the variance of noise σ_ϵ^2. The good results had slightly higher value of mean squared error that did not exceed $2 \cdot \sigma_\epsilon^2$. For the acceptable results the value of MSE lies in the range from $2 \cdot \sigma_\epsilon^2$ to $10 \cdot \sigma_\epsilon^2$. For deterministic systems (the case in which $\sigma_\epsilon^2 = 0$), the same boundaries as in the previous experiments were considered.

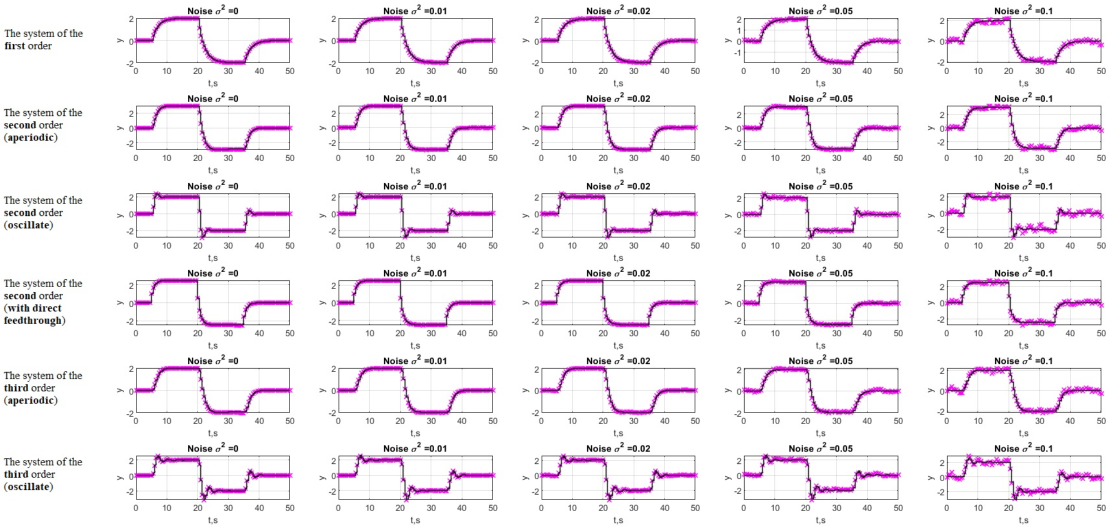
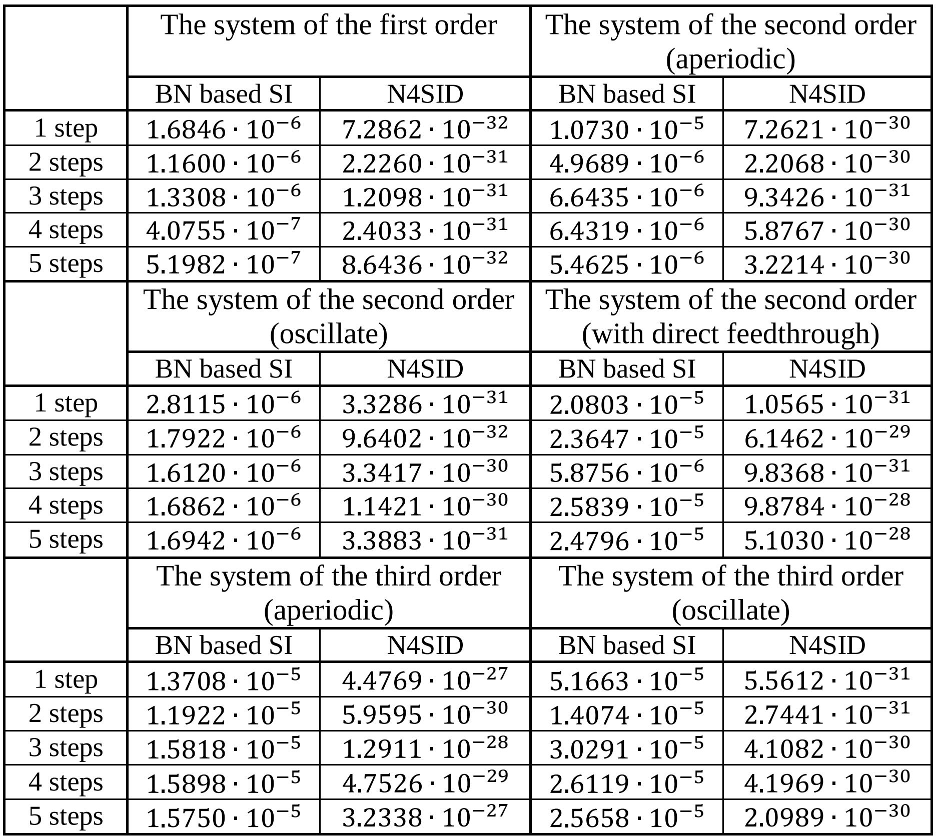


Figure 5.32: Results of BNSI for '2 steps' input signal

The learning procedure had the high ratio of successful experiments in all considered settings. It can be observed from the figure 5.30 that the precision decreased for higher value of variance, i.e. $\sigma_\epsilon^2 = 0.05$ and $\sigma_\epsilon^2 = 0.1$. However, increasing of the length of dataset used for identification increased the ratio of perfect results to the values comparable with the settings with lower variance of noise signal.

The best results from each identification bunch for the '1 step' input signal are shown on the figure 5.31. In most cases, the result of identification is perfect, except the aperiodic system of the third order, where learning algorithm approximated the dataset by an oscillate response. However, it should be pointed out that the dataset used in this setting for identification did not provide the unequivocal information regarding the behavior of the considered system (whether it is aperiodic or oscillate), even by visual assessment. Moreover, taking into account the small amount of available data (71 points), the high amount of unknown parameters and the high value of the variance of stochastic signal ($\sigma_\epsilon^2 = 0.1$), this result is not unusual. For longer sequences of input signal this problem did not persisted.

Table 5.8: Comparison of precision for deterministic systems



	The system of the first order		The system of the second order (aperiodic)	
	BN based SI	N4SID	BN based SI	N4SID
1 step	$1.6846 \cdot 10^{-6}$	$7.2862 \cdot 10^{-32}$	$1.0730 \cdot 10^{-5}$	$7.2621 \cdot 10^{-30}$
2 steps	$1.1600 \cdot 10^{-6}$	$2.2260 \cdot 10^{-31}$	$4.9689 \cdot 10^{-6}$	$2.2068 \cdot 10^{-30}$
3 steps	$1.3308 \cdot 10^{-6}$	$1.2098 \cdot 10^{-31}$	$6.6435 \cdot 10^{-6}$	$9.3426 \cdot 10^{-31}$
4 steps	$4.0755 \cdot 10^{-7}$	$2.4033 \cdot 10^{-31}$	$6.4319 \cdot 10^{-6}$	$5.8767 \cdot 10^{-30}$
5 steps	$5.1982 \cdot 10^{-7}$	$8.6436 \cdot 10^{-32}$	$5.4625 \cdot 10^{-6}$	$3.2214 \cdot 10^{-30}$
	The system of the second order (oscillate)		The system of the second order (with direct feedthrough)	
	BN based SI	N4SID	BN based SI	N4SID
1 step	$2.8115 \cdot 10^{-6}$	$3.3286 \cdot 10^{-31}$	$2.0803 \cdot 10^{-5}$	$1.0565 \cdot 10^{-31}$
2 steps	$1.7922 \cdot 10^{-6}$	$9.6402 \cdot 10^{-32}$	$2.3647 \cdot 10^{-5}$	$6.1462 \cdot 10^{-29}$
3 steps	$1.6120 \cdot 10^{-6}$	$3.3417 \cdot 10^{-30}$	$5.8756 \cdot 10^{-6}$	$9.8368 \cdot 10^{-31}$
4 steps	$1.6862 \cdot 10^{-6}$	$1.1421 \cdot 10^{-30}$	$2.5839 \cdot 10^{-5}$	$9.8784 \cdot 10^{-28}$
5 steps	$1.6942 \cdot 10^{-6}$	$3.3883 \cdot 10^{-31}$	$2.4796 \cdot 10^{-5}$	$5.1030 \cdot 10^{-28}$
	The system of the third order (aperiodic)		The system of the third order (oscillate)	
	BN based SI	N4SID	BN based SI	N4SID
1 step	$1.3708 \cdot 10^{-5}$	$4.4769 \cdot 10^{-27}$	$5.1663 \cdot 10^{-5}$	$5.5612 \cdot 10^{-31}$
2 steps	$1.1922 \cdot 10^{-5}$	$5.9595 \cdot 10^{-30}$	$1.4074 \cdot 10^{-5}$	$2.7441 \cdot 10^{-31}$
3 steps	$1.5818 \cdot 10^{-5}$	$1.2911 \cdot 10^{-28}$	$3.0291 \cdot 10^{-5}$	$4.1082 \cdot 10^{-30}$
4 steps	$1.5898 \cdot 10^{-5}$	$4.7526 \cdot 10^{-29}$	$2.6119 \cdot 10^{-5}$	$4.1969 \cdot 10^{-30}$
5 steps	$1.5750 \cdot 10^{-5}$	$3.2338 \cdot 10^{-27}$	$2.5658 \cdot 10^{-5}$	$2.0989 \cdot 10^{-30}$

For the ’2 steps’ input signal (the dataset with 101 points) the algorithm coped with learning task for all bunches. The best results from these identification experiments are presented on the figure 5.32. The results of identification using longer sequences of input and output signals were very similar to those obtained for the ’2 steps’ input signal and therefore their closer analysis will be omitted.

The results from identification were compared with the most commonly used method for identification using state space model, the N4SID. The comparison of mean squared errors for models obtained using both approaches for the deterministic setting ($\sigma_\epsilon^2 = 0$) can be found in the table 5.8. Even though the BN based SI provided successful results for all considered cases, the precision of obtained responses was considerably lower than for the N4SID method.

Table 5.9: Comparison of precision for stochastic systems

	The system of the first order		The system of the second order (aperiodic)	
	BN based SI	N4SID	BN based SI	N4SID
1 step	0.016414	0.020074	0.019648	0.037743
2 steps	0.019700	0.021140	0.027212	0.027258
3 steps	0.017567	0.018105	0.038318	0.038361
4 steps	0.016818	0.016829	0.025598	0.025852
5 steps	0.017852	0.018026	0.025548	0.025667
	The system of the second order (oscillate)		The system of the second order (with direct feedthrough)	
	BN based SI	N4SID	BN based SI	N4SID
1 step	0.034306	0.035553	0.013471	0.013444
2 steps	0.044670	0.044772	0.019570	0.019451
3 steps	0.045825	0.045890	0.021429	0.021184
4 steps	0.035270	0.035282	0.017429	0.017398
5 steps	0.041043	0.041324	0.016693	0.016747
	The system of the third order (aperiodic)		The system of the third order (oscillate)	
	BN based SI	N4SID	BN based SI	N4SID
1 step	0.012876	0.017054	0.031797	0.032348
2 steps	0.018935	0.019362	0.043625	0.044276
3 steps	0.016925	0.017193	0.028796	0.029184
4 steps	0.016942	0.016946	0.051995	0.052035
5 steps	0.019684	0.019720	0.057377	0.057405

However, for stochastic systems with the highest considered value of the noise variance ($\sigma_\epsilon^2 = 0.1$), the results obtained with BN based SI were more precise than N4SID for almost all considered cases, except the system with direct feedthrough, refer to the table 5.9. For other values of the noise variance, the BN based SI provided more precise results in many cases. In those, in which it did not, the precision was comparable to the one obtained with N4SID. Therefore, we can conclude that for considered simulated stochastic systems BN based SI provides the estimates which were at least comparable to those obtained with N4SID, but in many cases they were more precise, in particular for higher values of noise.

Therefore, we can conclude that Bayesian networks are an appropriate tool for identification of stochastic dynamic systems. While their behavior in experiments on deterministic systems was sometimes unpredictable, in the case of stochastic signals, it provides appropriate estimates and their precision can be improved by the increasing of the amount of data in a dataset used for identification. The chances to reach global optimum of log likelihood function can be increased remarkably by the several iterations of the learning procedure with different initialization values, the approach often used for iterative algorithms that optimize functions with multiple optima. The important question to solve in this context is the required amount of iterations, since the size of a bunch equal to 100 considered in all experiments in this thesis may be too high for practical applications. This question is addressed in the chapter 6.

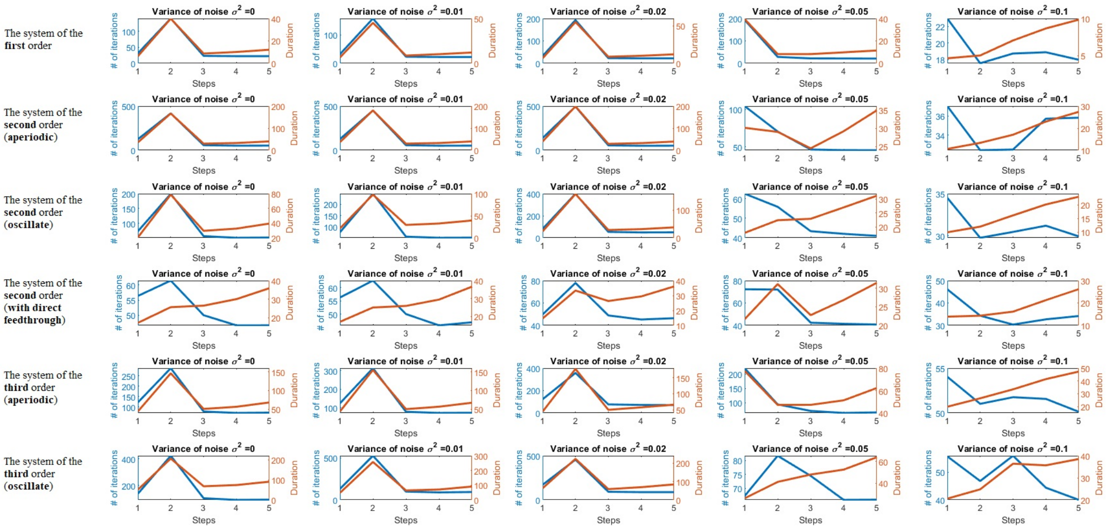


Figure 5.33: Numerical properties of identification for stochastic systems

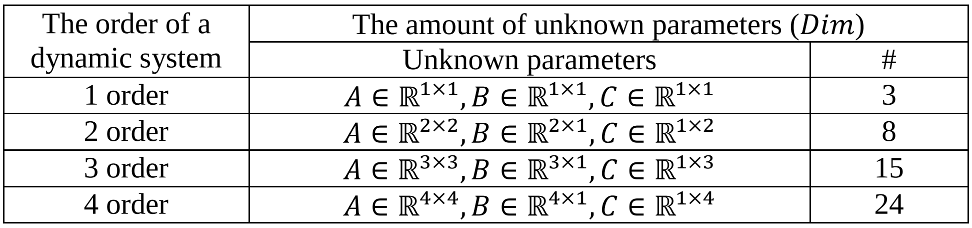
An important thing to assess are the numerical properties of the learning procedure during identification of stochastic dynamic systems. The average iteration amount required for EM algorithm to converge and the average duration of identification trial for each bunch are presented on the figure 5.33. As it can be observed from plots, both the average duration and the average amount of iterations decreased with increasing variance of noise for the majority of cases. In particular, for values of noise variance $\sigma_\epsilon^2 = 0.05$ and $\sigma_\epsilon^2 = 0.1$, this values drastically decreased for almost all considered dynamic systems. This unusual behavior indicates, that Bayesian networks can flourish the most in the identification of stochastic systems with higher noise.

5.7 Searching for the optimal system order

The simulated responses of stochastic systems were also used for exploring the proposed approach to order selection. In identification experiments provided so far in the thesis, it was assumed that the order of a considered system is known (we used the same order as the order of a transfer function from which a corresponding dataset was obtained). In many practical applications the appropriate order is not known in advance and therefore it has to be chosen during identification. The common approach to deal with this task is to use a penalized goodness-of-fit criterion, for example Akaike information criterion (AIC). We propose to use scoring functions from Bayesian network framework that are used for the choice of the most appropriate structure of a network in structure learning procedures. These scores assess how well the graph with estimated parameters fit training data. Consequently, the order selection approach is consistent with the parameter estimation approach.

Verifying of the proposed approach was provided for all previously considered stochastic dynamic systems. For this purpose, we used the available responses on the '3 steps' input signal with different influence of noise. The identification procedure was provided with three different values of system order for each case. For the aperiodic dynamic systems, the values of orders were set to $n = 1, 2, 3$, for the oscillate dynamic systems they were set to $n = 2, 3, 4$.

Table 5.10: Amount of independent parameters



The order of a dynamic system	The amount of unknown parameters (Dim)	
	Unknown parameters	#
1 order	$A \in \mathbb{R}^{1\times1}, B \in \mathbb{R}^{1\times1}, C \in \mathbb{R}^{1\times1}$	3
2 order	$A \in \mathbb{R}^{2\times2}, B \in \mathbb{R}^{2\times1}, C \in \mathbb{R}^{1\times2}$	8
3 order	$A \in \mathbb{R}^{3\times3}, B \in \mathbb{R}^{3\times1}, C \in \mathbb{R}^{1\times3}$	15
4 order	$A \in \mathbb{R}^{4\times4}, B \in \mathbb{R}^{4\times1}, C \in \mathbb{R}^{1\times4}$	24

We considered two most common scoring functions: likelihood (LL) score and Bayesian information criterion (BIC). The LL score is the value of log likelihood function for a graph accompanied with appropriate maximum likelihood estimate of parameters given a dataset. In BNT, this value is the last value of log likelihood stored by the EM algorithm.

When the likelihood score is used for learning of Bayesian network structure, it has in general the tendency to overfit training data. This results in the most complicated structure allowed by constraints (the structure with all allowed connections between nodes). The task of order selection is a little bit different, since we pick the most appropriate length of the state vector with the unchanged structure of underlying Bayesian network and therefore this behavior does not necessarily have to manifest itself.

Table 5.11: Values of likelihood score for different systems

System order	The likelihood score				
	$\sigma^2 = 0$	$\sigma^2 = 0.01$	$\sigma^2 = 0.02$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$
The system of the first order					
1 order	-42.4009	-44.0396	-46.2502	-74.0217	-133.6388
2 order	**-34.2469**	**-34.2257**	**-36.9023**	**-72.4196**	-133.5091
3 order	-35.6872	-50.3238	-42.1659	-87.6672	**-131.2771**
The system of the second order (aperiodic)					
1 order	-163.0102	-162.8812	-165.1009	-168.8811	-191.2833
2 order	**-41.2098**	**-43.6975**	**-49.4952**	**-93.9581**	-161.7276
3 order	-42.5962	-45.8280	-52.4308	-96.6489	**-160.7066**
The system of the second order (oscillate)					
2 order	**-34.7784**	**-38.9446**	**-51.0174**	-107.3522	-187.6088
3 order	-39.5889	-55.1811	-56.4349	-115.3526	-187.5080
4 order	-36.3149	-44.9766	-54.8316	**-104.5780**	**-185.4907**
The system of the second order (with direct feedthrough)					
1 order	-34.2857	-36.6002	-40.8416	-77.4575	-144.3546
2 order	-38.1817	-40.0226	-44.1678	-77.2657	**-140.4928**
3 order	**-31.9562**	**-33.9289**	**-38.8882**	**-75.2022**	-143.5438
The system of the third order (aperiodic)					
1 order	-165.4095	-165.6204	-165.1444	-173.0079	-188.3180
2 order	**-37.1941**	**-39.5093**	-45.3383	**-71.1908**	-131.8939
3 order	-38.8759	-40.7717	**-44.6917**	-76.6351	**-129.1940**
The system of the third order (oscillate)					
2 order	-83.3480	-86.4709	-97.4722	-128.1858	-182.5234
3 order	-56.3998	-52.2547	**-59.4173**	**-105.9906**	-159.4327
4 order	**-40.1203**	**-50.9019**	-61.1608	-110.0141	**-159.3487**

The BIC score is a penalized modification of the likelihood score. It is used more often for structure learning of Bayesian networks than the likelihood score due to its ability to trade-off model complexity versus accuracy using regularization techniques. The score penalizes the structures with higher amount of independent parameters, and asymptotically approaches the underlying structure of a considered Bayesian network for big training sets ($M \to \infty$). However, for small training sets the BIC score tends to underfit the dataset, because it penalizes the structure too hard. Therefore, it often chooses the simplest structure allowed by constraints The BIC score can be calculated using the following formula:

$$BIC = LL - \frac{logM}{2} \cdot Dim. \tag{5.4}$$

In the equation (5.4) LL is the value of the likelihood score, M is the number of training examples and Dim is the dimension of a graph expressed by the number of independent parameters in a graph. In the context of the system identification task, the dimension of a graph can be expressed by the number of unknown parameters in the state space model of a corresponding order. The amount of unknown parameters for SISO systems without direct feedthrough is listed in the table 5.10. For systems with direct feedthrough, we have to add one additional parameter for a dynamic system of arbitrary order ($\mathbb{D} \in \mathbb{R}^{(1 \times 1)}$). For MIMO systems, the amount of unknown parameters should be adjusted appropriately.

Table 5.12: Values of BIC score for different systems

System order	The BIC score				
	$\sigma^2 = 0$	$\sigma^2 = 0.01$	$\sigma^2 = 0.02$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$
The system of the first order					
1 order	**-49.7137**	**-51.3524**	**-53.5630**	**-81.3345**	**-140.9516**
2 order	-53.7477	-53.7265	-56.4031	-91.9204	-153.0099
3 order	-72.2512	-86.8878	-78.7299	-124.2311	-167.8411
The system of the second order (aperiodic)					
1 order	-170.3230	-170.1940	-172.4137	-176.1938	-198.5961
2 order	**-60.7106**	**-63.1983**	**-68.9960**	**-113.4589**	**-181.2284**
3 order	-79.1602	-82.3920	-88.9948	-133.2129	-197.2705
The system of the second order (oscillate)					
2 order	**-54.2792**	**-58.4453**	**-70.5182**	**-126.8530**	**-207.1096**
3 order	-76.1529	-91.7451	-92.9989	-151.9166	-224.0720
4 order	-94.8173	-103.4789	-113.3339	-163.0804	-243.9931
The system of the second order (with direct feedthrough)					
1 order	**-44.0361**	**-46.3506**	**-50.5920**	**-87.2078**	**-154.1050**
2 order	-60.1201	-61.9609	-66.1062	-99.2041	-162.4312
3 order	-70.9578	-72.9305	-77.8898	-114.2038	-182.5454
The system of the third order (aperiodic)					
1 order	-172.7223	-172.9332	-172.4572	-180.3207	-195.6308
2 order	**-56.6949**	**-59.0101**	**-64.8391**	**-90.6916**	**-151.3947**
3 order	-75.4399	-77.3357	-81.2557	-113.1991	-165.7580
The system of the third order (oscillate)					
2 order	-102.8488	-105.9717	-116.9730	-147.6866	-202.0242
3 order	**-92.9638**	**-88.8187**	**-95.9812**	**-142.5546**	**-195.9967**
4 order	-98.6227	-109.4042	-119.6631	-168.5165	-217.8511

The values of the likelihood score for the best result from each bunch of identification experiments are shown in the table 5.11. The maximum value of the likelihood score for each considered setting is highlighted with bold font. As it was already mentioned, when used for structure learning, the likelihood score tends to overfit a dataset and choose the most complicated structure allowed by constraints. In the context of order selection task, however, this behavior did not manifest.

The values of BIC score for the best result from each bunch of identification experiments are shown in the table 5.12. The maximum value of BIC score for each considered setting is highlighted with bold font. As it was already mentioned, when used for structure learning, the BIC score tends to underfit a dataset. In the context of order selection task, however, this behavior did not manifest. Moreover, the BIC score suggested to use the same order, as the order of the transfer function from which a dataset was generated for the system of the first order, both aperiodic and oscillate systems of the second order and the oscillate system of the third order for all considered values of the variance of noise signal. For the system of the second order with direct feedthrough and the aperiodic system of the third order, the BIC score managed to approximate the responses with the transfer functions of the order one less than the original value of an order (the first and the second respectively).

Table 5.13: Orders suggested by scoring functions and AIC

	System of the first order			System of the second order (aperiodic)			System of the second order (oscillate)		
	LL score	BIC score	AIC score	LL score	BIC score	AIC score	LL score	BIC score	AIC score
$\sigma^2 = 0$	2	1	1	2	2	2	2	2	3
$\sigma^2 = 0.01$	2	1	1	2	2	3	2	2	2
$\sigma^2 = 0.02$	2	1	1	2	2	2	2	2	2
$\sigma^2 = 0.05$	2	1	2	2	2	2	4	2	2
$\sigma^2 = 0.1$	3	1	1	3	2	2	4	2	2
	System of the second order (with direct feedthrough)			System of the third order (aperiodic)			System of the third order (oscillate)		
	LL score	BIC score	AIC score	LL score	BIC score	AIC score	LL score	BIC score	AIC score
$\sigma^2 = 0$	3	1	2	2	2	3	4	3	3
$\sigma^2 = 0.01$	3	1	3	2	2	3	4	3	3
$\sigma^2 = 0.02$	3	1	2	3	2	3	3	3	3
$\sigma^2 = 0.05$	3	1	2	2	2	3	3	3	3
$\sigma^2 = 0.1$	2	1	2	3	2	3	4	3	3

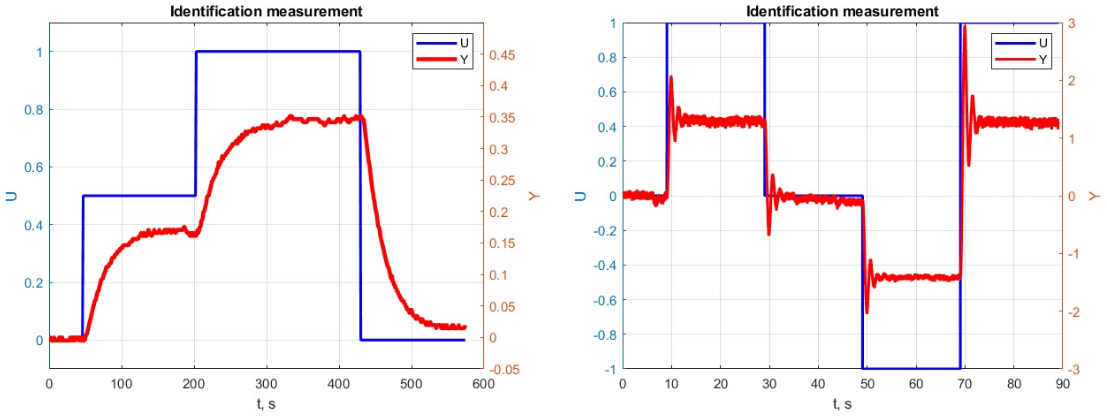
The efficiency of scoring functions was compared with Akaike's information criterion (AIC). This criterion was implemented using aic function from the System identification toolbox in MATLAB.

The comparison of orders suggested by scoring functions from Bayesian network framework and from AIC are shown in the table 5.13. It can be seen that while the LL score was prone to overfitting, the BIC score chose smaller value of system order for two out of six considered dynamic systems. In other settings, the behaviour of the latter was similar to the behaviour of AIC. However, in some settings, AIC tended to choose higher order than the order of a transfer function from which a corresponding dataset was generated.

6 Verification of the proposed approach on real systems

All experiments described so far in the thesis were provided on simulated responses of dynamic systems. The noise signals were generated using a random number generator of normally distributed variables. The generated sequences were added to the output signal and each state variable.

Verification of the proposed approach to system identification required datasets influenced by real noise that does not necessarily come from normal distribution in practical measurements. Therefore, we searched for physical systems (aperiodic and oscillate) that can be used for verification. The main requirements were: the linearity of considered systems, the presence of noticeable noise and easy manipulation. The laboratory equipment used for teaching control theory classes at the Technical university of Liberec fulfilled all these requirements. The first equipment is a heating device, the second equipment is a DC motor connected to a dynamo with an elastic clutch. Both dynamic systems have linear behavior in the proximity to an operational point and therefore the tools for analysis of linear dynamic systems



(a) Aperiodic system

(b) Oscillate system

Figure 6.1: Identification measurements provided on laboratory equipment

can be used if the appropriate preprocessing (excluding of the response representing the transfer to an operational point and moving of remaining responses to an operational point) is provided. The identification measurement for the first dynamic system is shown on the figure 6.1a, the behavior of a system is aperiodic. The identification measurement for the second dynamic system is shown on the figure 6.1b, the system has noticeable oscillations.

Table 6.1: Design choices in identification procedure

Dynamic system	Considered sampling rates	Considered orders
The aperiodic system	$dt = [1 \quad 2 \quad 5 \quad 10 \quad 20]$	$n = [1 \quad 2 \quad 3]$
The oscillate system	$dt = [0.05 \quad 0.1 \quad 0.2 \quad 0.5 \quad 1]$	$n = [2 \quad 3 \quad 4]$

The efficiency of Bayesian network based approach to system identification for different values of sampling rate was explored. For this purpose, five different sampling rates were chosen for each dynamic system. We assume that no prior information regarding the order of considered dynamic systems is available and therefore the identification using state space representation of different orders is provided. The information regarding considered sampling rates and system orders can be found in the table 6.1. Identification measurements were reduced with respect to chosen sampling rates. Resulting datasets for the aperiodic system are shown on the figure 6.2 and for the oscillate system on the figure 6.3. These datasets were used in the identification experiments described in the following sections.

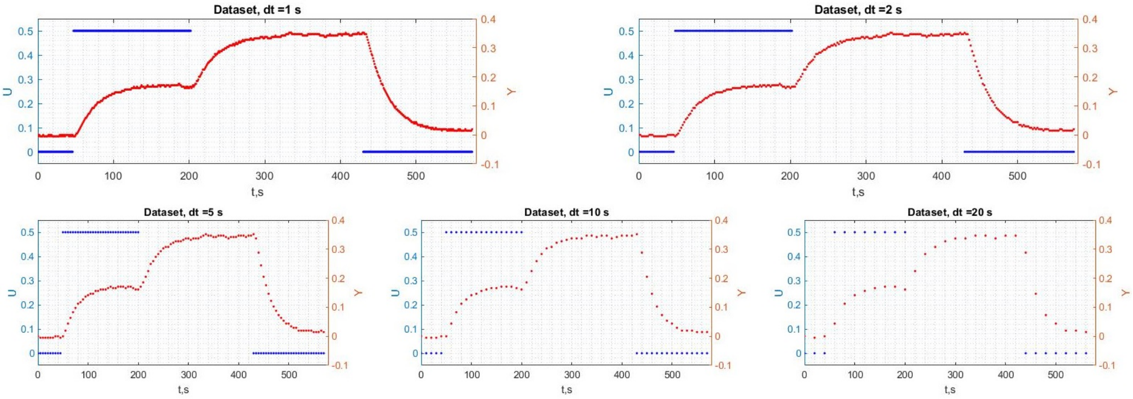


Figure 6.2: Datasets used for identification of the aperiodic system

The efficiency of Bayesian network tools for order selection was explored in looking for the most appropriate order of a final description. For this purpose, two scoring functions (likelihood score and BIC score) were utilized.

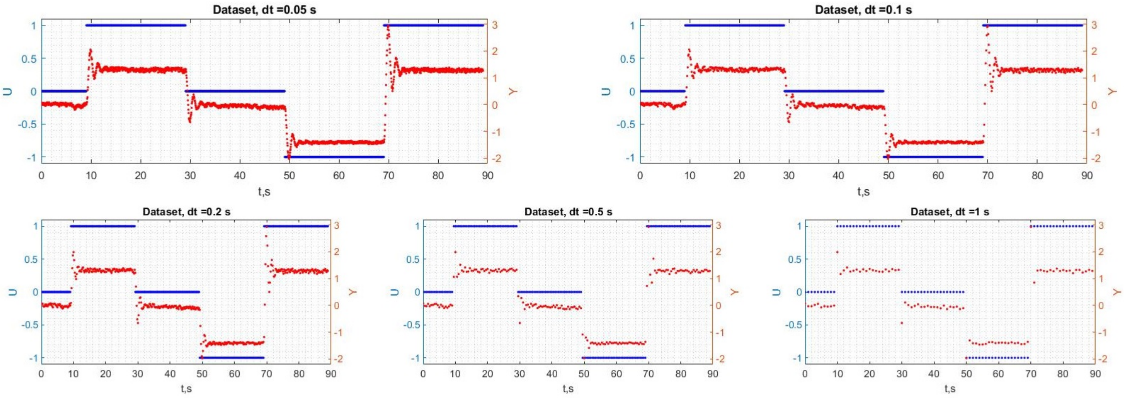


Figure 6.3: Datasets used for identification of the oscillate system

For this experiment, 15 bunches of identification trials with 100 identification trials in each bunch were provided for each considered dynamic system. The identification procedure was provided according to the learning scenario 5 (refer to the subsection 5.3.5) with initial values of parameters sampled from the standard normal distribution (the default setting, refer to the section 5.5). For each bunch of identification trials, the best result was chosen, i.e. the one with the lowest value of the mean squared error.

6.1 Identification of the aperiodic dynamic system

Results of Bayesian network based system identification for the aperiodic dynamic system are presented on the figure 6.4. The corresponding value of a mean squared error can be found in the title of each plot. The results of identification using N4SID method are presented on the figure 6.5. These results were obtained using the function n4sid from System Identification Toolbox in MATLAB. Identification with default arguments ended up with several unsuccessful results and therefore the stability of resulting systems was enforced.

The comparison of mean squared errors for Bayesian networks and N4SID shows that the precision of obtained estimates using Bayesian networks is comparable with N4SID and in particular cases (for higher values of a sampling rate) is slightly better.

The disadvantage of Bayesian networks in comparison with N4SID is the dura-

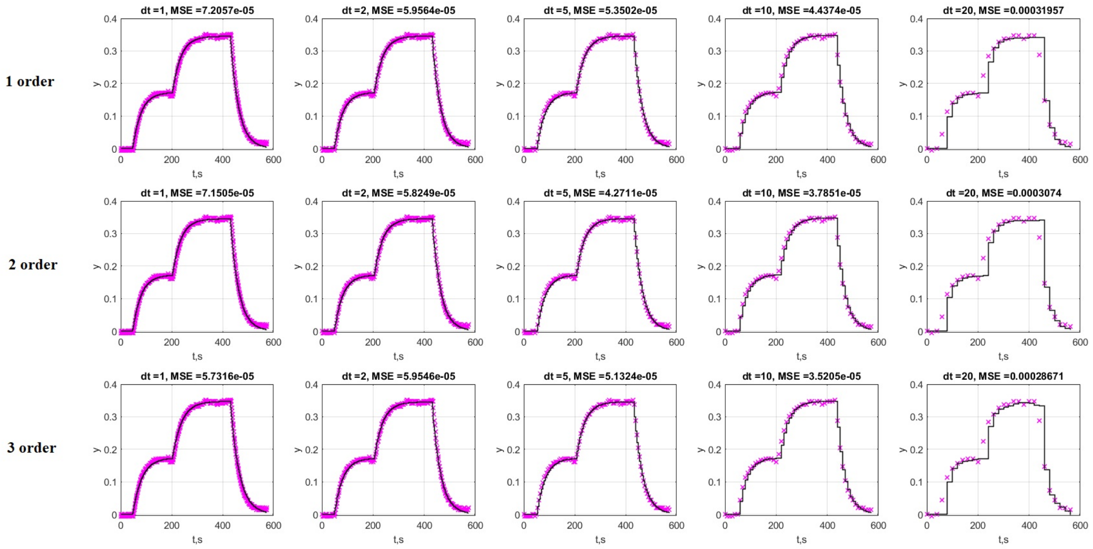


Figure 6.4: Results of BNSI for the aperiodic system

tion of identification procedure. Since initialization of initial values of parameters is required for the EM algorithm used for learning of Bayesian networks, it is important to explore the required amount of initializations. For this purpose, for each considered setting the minimal value of the cost function J (mean squared error) was calculated for different amount of initializations. The resulting plots are presented on the figure 6.6. It can be seen, that for the considered system the minimum

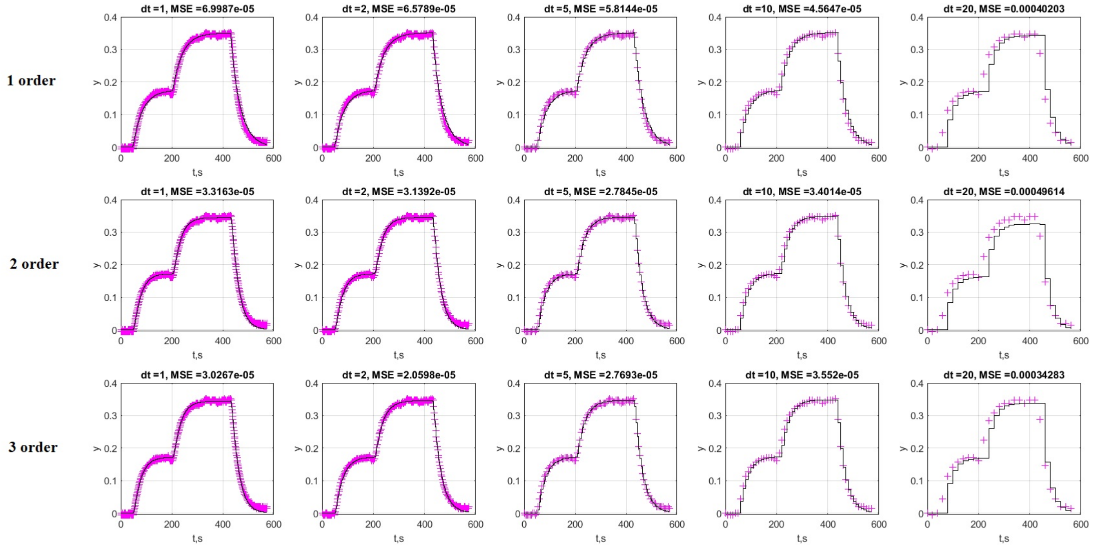


Figure 6.5: Results of identification using N4SID for the aperiodic system

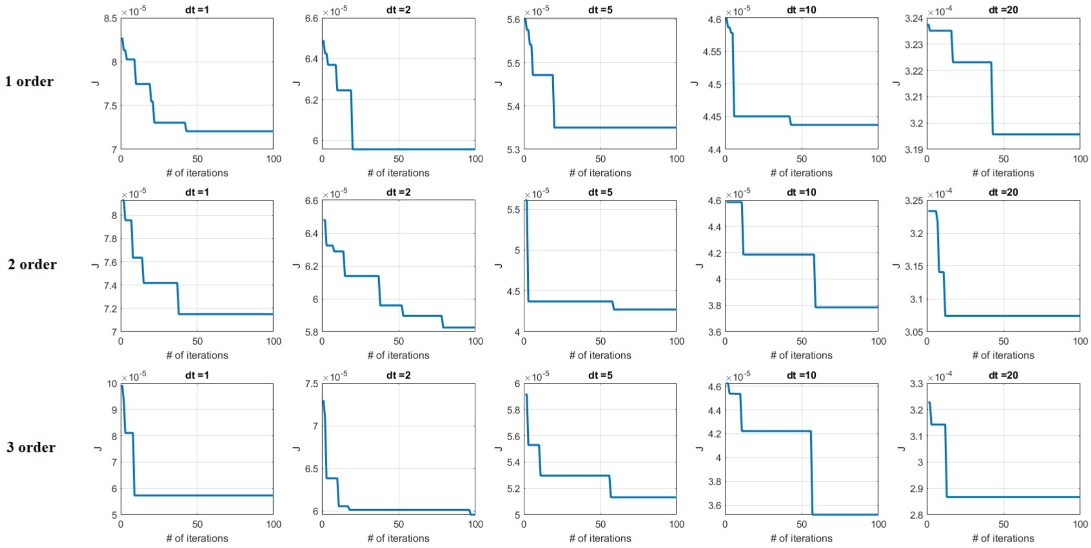


Figure 6.6: Precision of results for different amount of initializations for the aperiodic system

value of the cost function was quite low even for the small amount of initializations. However, the increasing of the amount of initializations provided slightly better estimates. Therefore, it can be concluded that the amount of required iterations depends on the precision requirements.

Table 6.2: Values of scoring functions for the aperiodic system

Order	Sampling rate				
	$dt = 1$	$dt = 2$	$dt = 5$	$dt = 10$	$dt = 20$
The likelihood score					
1 order	184.8600	93.8935	37.8029	19.8160	**9.1741**
2 order	185.0481	**93.2992**	**37.9647**	**19.8892**	8.9427
3 order	**193.5730**	93.1944	37.6531	19.8426	8.7602
The BIC score					
1 order	**175.3284**	**85.3991**	**30.6855**	**13.7253**	**4.1232**
2 order	159.6307	70.6474	18.9850	3.6475	-4.5265
3 order	145.9153	50.7222	2.0661	-10.6108	-16.4946

Order selection was provided using likelihood score and BIC score, their values are given in the table 6.2. The maximum values of scores for each considered case are highlighted with bold font. The likelihood score suggested to use higher order for smaller sampling rate, but did not have the general tendency to overfit training data. The BIC score suggested to use the description of the first order for each considered sampling rate.

Table 6.3: Suggested orders for the aperiodic system

System order	Orders suggested by scores				
	$dt = 1$	$dt = 2$	$dt = 5$	$dt = 10$	$dt = 20$
Likelihood score	3	2	2	2	1
BIC score	1	1	1	1	1
Common scores	3	3	2	2	1

The system orders suggested by scoring functions were compared to the orders suggested by the AIC, refer to the table 6.3. The AIC proposed to use the description of the third order for lower values of sampling rate, and the value of suggested order decreased with increasing sampling rate.

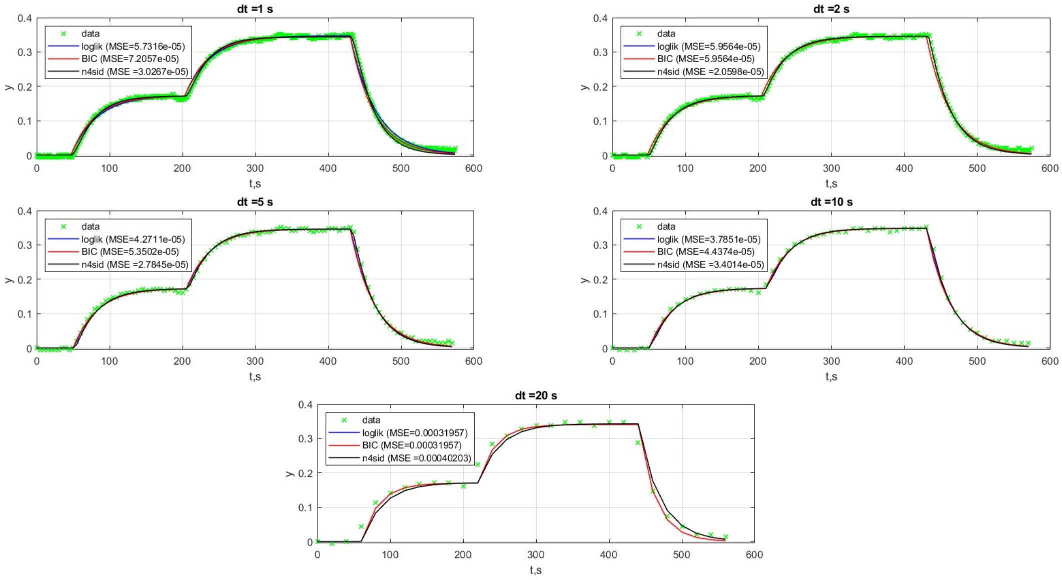


Figure 6.7: Comparison between BNSI and N4SID with optimal order for the aperiodic system

To access the efficiency of system identification procedure including order selection, the comparison between results obtained from BNSI with optimal order and N4SID with optimal order was provided. Responses from corresponding models for each considered sampling rate are shown on the figure 6.7. In the legend, the values of mean squared error are listed. It can be seen, that responses obtained from the learning procedure using both scoring functions provided slightly worse, but comparable results with N4SID method in all settings except the one with sampling rate $dt = 20s$, in which BNSI provided more precise model.

6.2 Identification of the oscillate dynamic system

Results of Bayesian network based system identification for the oscillate dynamic system are presented on the figure 6.8. The corresponding value of a mean squared error can be found in the title of each plot.

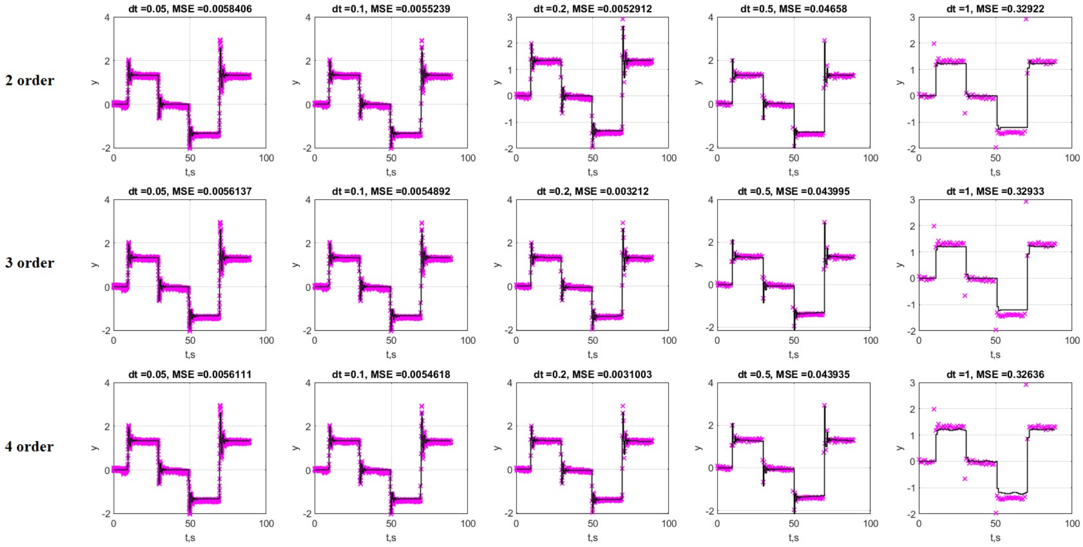


Figure 6.8: Results of BNSI for the oscillate system

In general, the oscillate dynamic systems are more complicated to identify than the aperiodic systems and are more prone to unsuccessful results. In particular, the issues are caused by the fact that the identification procedure has to distinguish between the natural oscillations of considered system and the influence of noise. If it fails in this task, it approximates the oscillate behavior by the aperiodic description of dynamic systems. This issue becomes more challenging if the chosen sampling rate is not appropriate. Too high value of sampling rate causes the identification procedure to misclassify oscillations as noise (see results of different order for $dt = 1$ on the figure 6.8), too low value sampling rate may cause numerical problems since the original discrete description of dynamic systems contains very small values of coefficients (their value may even be comparable with the magnitude of a noise signal).

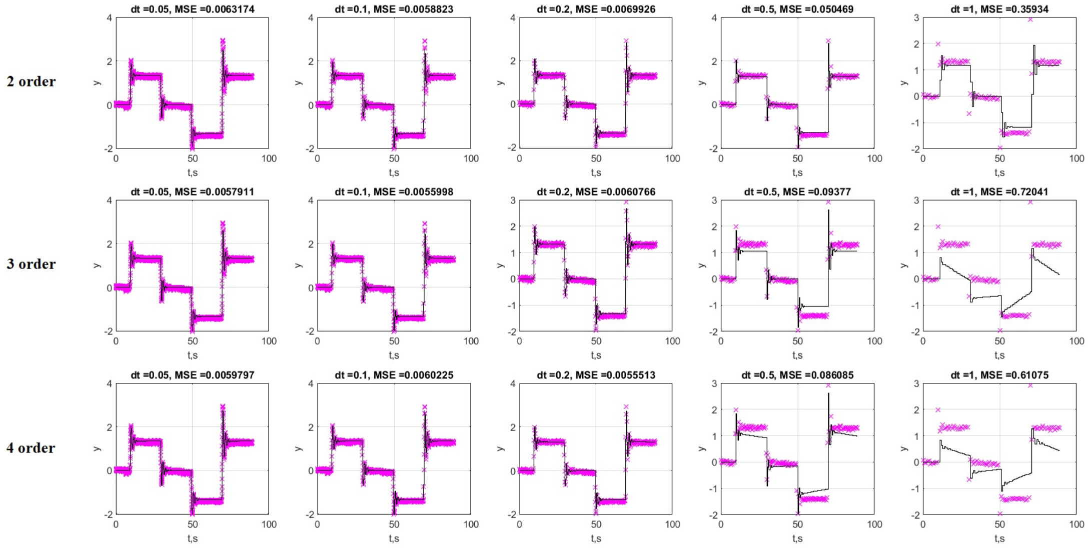


Figure 6.9: Results of identification using N4SID for the oscillate system

The learning algorithm succeeded to get appropriate behavior for all dynamic systems with sampling rate $dt < 1$. The N4SID method was not as successful, see figure 6.9. During identification procedure, the stable behavior of resulting descriptions was enforced to eliminate non-stable results obtained from the N4SID algorithm. The precision of obtained estimates in the terms of mean squared error for Bayesian networks was better for all considered settings.

Analogously to the aperiodic system, the amount of iterations required to obtain satisfying precision was analyzed. The minimum value of the cost function for different amount of initializations in each setting is shown on the figure 6.10. Analogously to the aperiodic system, the precision of obtained results was good even for the small amount of iterations, and it was decreasing with increasing amount of initializations.

The last task that had to be addressed for the oscillate system was the selection of the optimal order of resulting description. Analogously to the aperiodic system, order selection was provided using likelihood score and BIC score. Their values for each considered case are shown in the table 6.4, the maximal values of scores for each sampling rate are highlighted with bold font. They correspond to the suggested value of system order.

The likelihood score did not show the general tendency to overfit the training data. For three out of five sampling rates it suggested to use the highest considered order (fourth order), but for sampling rate $dt = 0.1s$ the optimal order was the lowest one (second order), and for the sampling rate $dt = 0.2s$ the likelihood score

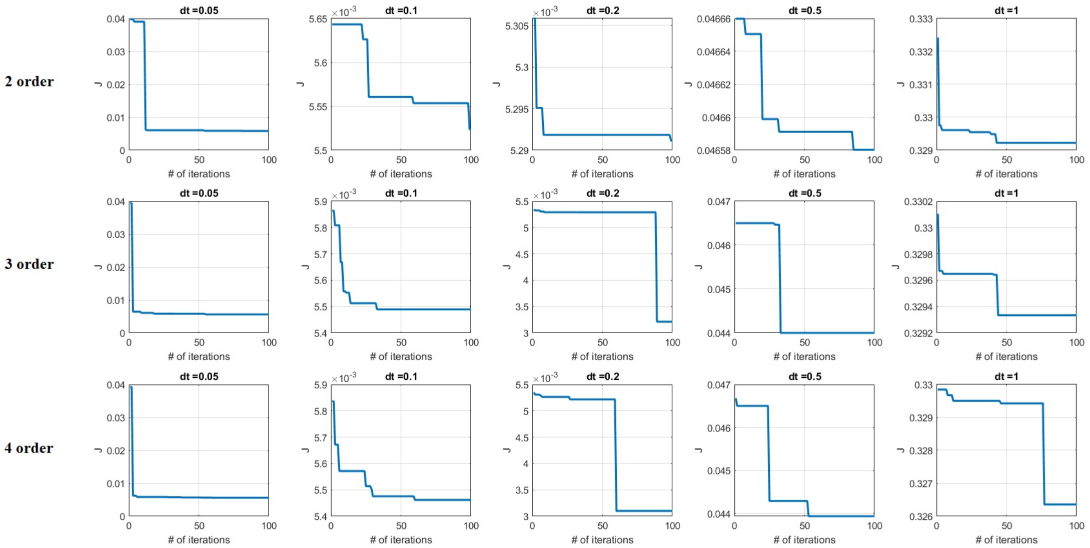


Figure 6.10: Precision of results for different amount of initializations for the oscillate system

indicated, that the third order is the best one to pick. The BIC score proposed to use the description of the second order in all cases except the one with sampling rate $dt = 0.2s$, in which the description of the third order was suggested.

Table 6.4: Values of scoring functions for the oscillate system

Order	Sampling rate				
	$dt = 0.05$	$dt = 0.1$	$dt = 0.2$	$dt = 0.5$	$dt = 1$
The likelihood score					
2 order	-323.4572	**-208.7061**	-111.7674	-202.4485	-187.2146
3 order	-307.8886	-208.7803	**-57.7851**	-198.1311	-187.2997
4 order	**-303.9422**	-218.8368	-63.1477	**-198.0896**	**-186.8842**
The BIC score					
2 order	**-353.3992**	**-235.8755**	-136.1687	**-223.1981**	**-205.2138**
3 order	-364.0298	-259.7228	**-103.5375**	-237.0365	-221.0483
4 order	-393.7681	-300.3449	-136.3515	-260.3382	-240.8819

The system orders suggested by scoring functions were compared to the orders suggested by AIC, refer to the table 6.5. Similarly to the LL and BIC scores, the AIC proposed to use different values of system order for different sampling rates.

Table 6.5: Suggested orders for the oscillate system

System order	Orders suggested by scores				
	$dt = 0.05$	$dt = 0.1$	$dt = 0.2$	$dt = 0.5$	$dt = 1$
Likelihood score	4	2	3	4	4
BIC score	2	2	3	2	2
Common scores	4	2	4	3	3

To access the efficiency of system identification procedure including order selection, the comparison between results obtained from BNSI with optimal order and N4SID with optimal order was provided. Responses from corresponding models for each considered sampling rate are shown on the figure 6.11. The responses are reduced for better visualization. In the legend, the values of mean squared error are listed. It can be seen that responses obtained from the learning procedure using both scoring functions provided more precise results than N4SID method in the terms of mean squared error in all considered settings. Although, BNSI failed to distinguish oscillate behavior of a considered system for sampling rate $dt = 1s$, N4SID completely failed to properly identify the system in this setting.

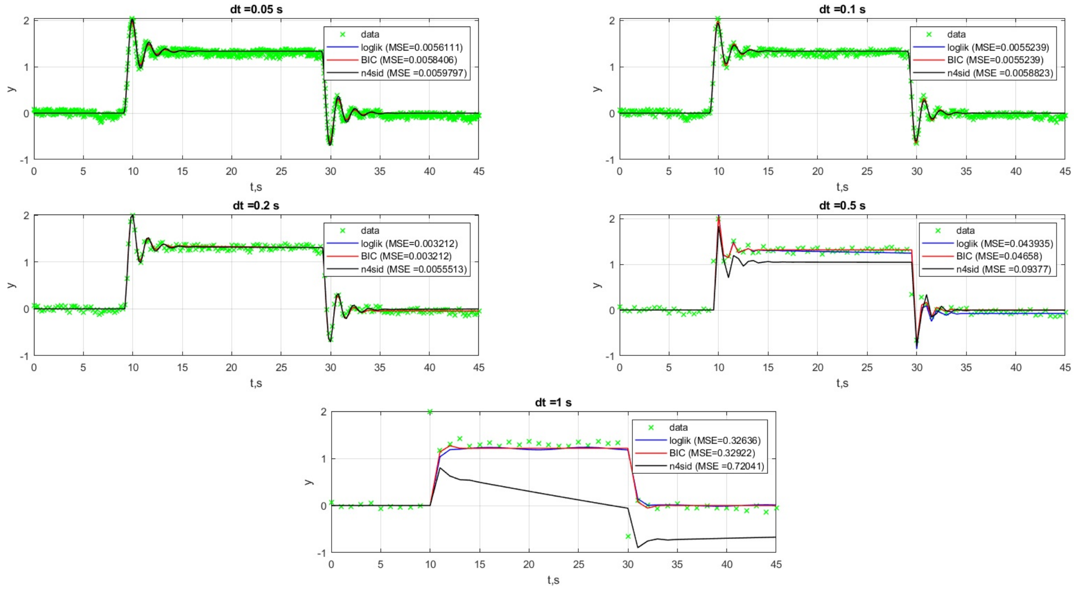


Figure 6.11: Comparison between BNSI and N4SID with optimal order for the oscillate system

7 Contributions of the thesis and directions for further research

On the basis of provided experiments, it can be concluded, that the presented approaches to parameter estimation and order selection using Bayesian networks can be used for system identification of dynamic systems. The thesis contributes both to a scientific field (section 7.1) and to practice (section 7.2). The directions for further research are proposed in the section 7.3.

7.1 Contributions to a scientific field

The thesis brings the following contributions to a scientific field:

- A new approach to the identification of dynamic systems, Bayesian network based system identification (BNSI), was proposed and verified in this thesis (refer to the chapter 5 and the chapter 6).
- The new approach was enriched by the order selection procedure proposed and verified in the section 5.7).
- Models of dynamic systems based on difference equation (section 4.1) and state space representation (section 4.2) were formulated using Bayesian network framework.
- Bayesian network based state observers were proposed in the section 4.3.
- Extensive literature review is provided in the chapter 1. It includes the overview of the most prominent system identification techniques (section 1.1), the synopsis of the genesis and development of Bayesian networks with the overview of their recent implementations (section 1.2) and the state of the art in the interconnection between Bayesian networks and control systems engineering (section 1.3).

7.2 Contributions to practice

The thesis brings the following contributions to practice:

- A general software-independent methodology of Bayesian network based system identification is formulated in the section 5.1. Implementation of BNSI for state space model of dynamic systems in BNT is discussed in details in the subsection 5.2.1.

- The Bayesian network based models of dynamic systems presented in the chapter 4 can be used for system identification, monitoring, state estimation and control of technological processes

- The literature review contains references to the plenty of practical implementations of Bayesian networks in control systems engineering (section 1.3) and beyond (section 1.2).

7.3 Directions for further research

Since this thesis proposes a new approach to system identification, we can suggest many directions for further research in this area. Some of them follow from the experiments provided in the scope of this thesis, other suggestions are more general.

The suggestions that follow from the conducted experiments aim to decrease the time of identification cycle and the amount of required initializations of initial parameters. The former challenge can be addressed by the implementation of the learning procedure using context-specific functions in other programming language than MATLAB for providing more efficient code (e.g. C/C++). For the latter issue, more systematic study of the influence of the mean of distribution for initial parameters in the EM algorithm can be beneficial. Another way for addressing this issue is the implementation of different algorithms for parameter estimation in Bayesian networks.

General suggestions for further research are connected with the extensions of Bayesian network based models of dynamic systems presented in this thesis.

One of the strong assumptions made for models of dynamic systems discussed in this thesis was the linearity of considered systems. It is important to point out, that Bayesian networks do not put any restrictions on the type of used connections between random variables [1]. However, available software packages mostly support only linear type of interconnection and hence using of other (possibly nonlinear) functions requires the adaptation of inference and learning algorithms. Less demanding way to incorporate nonlinear dependencies between random variables is linear approximation. This approach to modelling of BN-based dynamic models was proposed by R. Deventer et al. [213]. Authors show, how a hybrid Bayesian networks (a network with both discrete and continuous nodes) designed on the basis of a piecewise linear approximation with several base points can serve for this purpose. For more information on this approach refer to [12].

The normality of all considered variables is another strong assumption made for models in this thesis. Since it may not be always fulfilled in practical applications, it is important to discuss, how it can be relaxed. Analogously to the previous assumption, this one was also caused by the software restrictions and not by the limitations of Bayesian networks. Theoretically, Bayesian networks can describe random variables with arbitrary distribution, however the representatives of exponential family will be the easiest for implementation, refer to [1]. For less demanding approach we can again use approximations. Since nearly any continuous distribution can be approximated by the mixture of sufficient amount of Gaussian distributions [27], we can use mixtures instead of normally distributed variables. The mixture of Gaussian distributions can be elegantly implemented by a hybrid Bayesian network, but this approach can bring additional computational burden.

Models presented in this thesis can be enriched to model wider range of dynamic systems. For example, continuous-time dynamic systems can be modelled by a continuous time Bayesian network (CTBN), presented by U. Nodelman et al. [228, 229]. A CTBN is an extension of a DBN based on continuous-time Markov processes that describe a model evolving over continuous time. In addition, this model does not put any restrictions on acyclicity and hence feedback loop can be implemented explicitly. The combination of a continuous-time and a discrete-time dynamic Bayesian network, called a hybrid-time Bayesian network, was proposed by M. Liu et al. [230].

For demanding technological processes further extensions of the DBN structure can be used, e.g. adaptive BNs [231], time-varying BNs [232], infinite BNs [233].

Another way to cope with the complexity of considered dynamic systems modelled by a dynamic Bayesian networks is to divide their structure into factors that are treated separately. This approach is used particularly for efficient tracking of technological processes. The decomposition of a BN structure can be provided by Boyen-Koller algorithm [234], distributed decentralized extended Kalman filter [235], heuristic techniques [236], factored particle filtering [237] or factoring using structural observability [238].

8 Conclusion

Bayesian networks have become one of the main tools for reasoning under the influence of uncertainties in artificial intelligence community. They are particularly useful for situations that incorporate partial expert knowledge and partially observed data. Their further advantages are modularity and the ability to generalize over a wide set of models. The former follows from the separation of representation (principles of model composition) and reasoning (inference and learning algorithms) and the latter allows implementation of well-known models and their combinations using Bayesian networks framework. In combination, these advantages create the huge field of potential implementations for Bayesian networks. Most of the algorithms in Bayesian network framework are not application-specific and hence they can be used for broad range of practical applications. Since the framework is constantly evolving, new techniques can be adopted in control systems engineering, provided that the bridging between them has been created.

The review over available literature (section 1.3) has proven, that the providing of this bridging is the ongoing topic of research. In particular, Bayesian networks were successfully used for monitoring and control of dynamic systems. The interconnection between Bayesian networks and system identification, however, was not addressed in available literature, and therefore, bridging of these two scientific branches was the main aim of this thesis.

Using of Bayesian networks for reformulation of difference equation and state space representation of dynamic systems was proposed by R. Deventer [12]. This modelling paradigm was adopted in this thesis. On its basis, models of dynamic systems based on static and dynamic Bayesian networks were designed (section 4). In addition, the potential of state space based Bayesian network structures as the new type of state observers was discussed in the section 4.3. These structures can be used separately or in combination with traditional state observers (i.e. Luenberger observer and Kalman filter).

The thesis proposes a new approach to system identification using algorithms from the Bayesian network framework in the section 5.1. The efficiency of Bayesian network based system identification procedure was studied for the state space based Bayesian network structure using the simulated deterministic responses of the most popular types of dynamic systems. The influence of different settings in the learning procedure on the required computation time and the precision of resulting estimates was explored (sections 5.3, 5.4 and 5.5).

Firstly, the influence of incorporating the expert knowledge in the form of variances of input and output signals and correlation matrix for state variables was explored in the section 5.3. The precision, duration and numerical stability of calculations differed depending on a setting and the type of a dynamic system. The optimal setting, which was chosen for further experiments, was the one with diagonal type of covariance matrix (refer to the subsection 5.3.5). Although, this setting was not the fastest and was not the most precise for some types of dynamic systems (i.e. the system of the first order and the system with direct feedthrough), it was numerically stable and its precision did not depend on the amount of available data. For all considered systems it provided perfect results in more than a half of iterations.

Secondly, the influence of incorporating the expert knowledge in the form of partial information regarding parameters was explored in the section 5.4. The fixing of parameters had positive effect on the precision for the system of the first order and the system with direct feedthrough. For other systems, the precision of obtained estimates decreased. The other approach explored in this section was to sample some parameters from standard normal distribution and fix them (exclude them from learning). The increase in precision was observed only for the system of the first order in this setting.

Finally, the influence of initial values of parameters was explored (section 5.5). Setting of their values to zero did not provide correct parameter estimates for all considered dynamic systems and therefore they were sampled from a random generator. Two types of generators (uniform and normal) with different parameters (mean, variance and range) were used. Experiments showed, that the type of used generator and the value of variance (for normal distribution) or range (for uniform distribution) had little to no effect on the precision of obtained estimates. The influence of mean on the precision of obtained estimates was more noticeable, however, it differed for the different types of dynamic systems and could not be generalized. Therefore, the type of chosen distribution was standard normal distribution.

Verifying of the designed Bayesian network based system identification procedure was provided on simulated responses of considered dynamic systems enriched by the influence of white noise with different variance (section 5.6). The precision of obtained results was compared with traditional method used for system identification of state space models (N4SID). For deterministic systems, Bayesian network based system identification was less precise than N4SID, but for stochastic systems its precision was at least comparable and in many cases better than the precision of N4SID.

Order selection approach based on the tools from Bayesian network framework was proposed and verified in the section 5.7. The efficiency of two scoring functions was explored. Likelihood score had a tendency to choose higher order, than the order of system from which data were simulated, but its penalized modification, Bayesian information score, suggested the same order for the majority of considered dynamic systems. The exceptions were the system of the second order with direct feedthrough and the aperiodic system of the third order. In both cases, Bayesian information score suggested to use the description of order one less than the original value. Obtained results were compared to the orders suggested by Akaike's information criterion.

The proposed approach to system identification including proposed order selection technique was verified on the responses of real dynamic systems (section 6). The efficiency of BNSI was explored for different values of sampling time. For the aperiodic dynamic system the precision of obtained models was comparable with the models estimated using N4SID. For the oscillate dynamic system, BNSI overcame N4SID in precision of obtained estimates for all considered values of sampling rate.

Bibliography

[1] Koller, D. and Friedman, N., *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

[2] Isermann, R. and Münchhof, M., *Identification of Dynamic Systems. An Introduction with Applications*. Springer-Verlag Berlin Heidelberg, 2011.

[3] Maybeck, P. S., *Stochastic Models, Estimation, and Control. Volume 1*. New York: Academic Press, Inc., 1979.

[4] Fabre, E., "Bayesian Networks of Dynamic Systems," Ph.D. dissertation, Université de Rennes, 2007.

[5] Lerner, U., Moses, B., Scott, M. *et al.*, "Monitoring a Complex Physical System using a Hybrid Dynamic Bayes Net," in *Proceedings of 18th Conference on Uncertainty in Artificial Intelligence*, 2002, pp. 301–310.

[6] Deventer, R., Denzler, J., and Niemann, H., "Control of Dynamic Systems Using Bayesian Networks," in *Proceedings of the IBERAMIA/SBIA*, 2002, pp. 33–39.

[7] Verron, S., Tiplica, T., and Kobi, A., "Multivariate control charts with a bayesian network," in *4th International Conference on Informatics in Control, Automation and Robotics*, 2007, pp. 228–233.

[8] Welch, R. L. and Smith, C., "Bayesian Control for Concentrating Mixed Nuclear Waste," in *Proceedings of the 15th conference on Uncertainty in artificial intelligence*, 1999, pp. 663–669.

[9] Lerner, U., Parr, R., Koller, D., and Biswas, G., "Bayesian Fault Detection and Diagnosis in Dynamic Systems," in *Proceedings of the 17th National Conference on Artificial Intelligence*, 2000, pp. 531–537.

[10] Atoui, M. A., Verron, S., and Kobi, A., "Fault detection with Conditional Gaussian Network," *Engineering Applications of Artificial Intelligence*, vol. 45, pp. 473–481, 2015.

[11] Roweis, S. and Ghahramani, Z., "A Unifying Review of Linear Gaussian Models," *Neural computation*, vol. 11, no. 2, pp. 305–345, 1999.

[12] Deventer, R., "Modeling and Control of Static and Dynamic Systems with Bayesian Networks," Ph.D. dissertation, Universität Erlangen - Nürnberg, 2004.

[13] Murphy, K. P., "Dynamic Bayesian Networks: Representation, Inference and Learning," Ph.D. dissertation, University of California, Berkley, 2002.

[14] Ljung, L., "Perspectives on system identification," *Annual Reviews in Control*, vol. 34, no. 1, pp. 1–12, 2010.

[15] Tidriri, K., Chatti, N., Verron, S., and Tiplica, T., "Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges," *Annual Reviews in Control*, vol. 42, pp. 63–81, 2016.

[16] Ljung, L., *System Identification. Theory for the User. Second Edition.* New Jersey: Prentice Hall PTR, 1999.

[17] Söderström, T. and Stoica, P., *System Identification.* Prentice Hall, 1989.

[18] Layer, E. and Tomczyk, K., *Measurements, Modelling and Simulation of Dynamic Systems.* Springer-Verlag Berlin Heidelberg, 2010.

[19] Van Overschee, P. and De Moor, B., *Subspace Identification for Linear Systems. Theory - Implementation - Applications.* Springer US, 1996.

[20] Zadeh, L. A., "On the identification problem," *IRE Transactions on Circuit Theory*, vol. 3, no. 4, pp. 277–281, 1956.

[21] Goodwin, G. C. and Payne, R. L., *Dynamic System Identifications: Experiment Design and Data Analysis.* New York: Academic Press, Inc., 1977, vol. 136.

[22] Walter, E. and Pronzalo, L., *Identification of Parametric Models from Experimental Data.* Springer-Verlag Berlin Heidelberg New York, 1997.

[23] Keesman, K. J., *System Identification: An Introduction.* London: Springer-Verlag London Limited, 2011.

[24] Chen, B., Zhu, Y., Hu, J., and Príncipe, J. C., *System Parameter Identification: Information Criteria and Algorithms.* Elsevier, 2013.

[25] Raol, J. R., Girija, G., and Singh, J., *Modelling and Parameter Estimation of Dynamic Systems.* London: The Institution of Engineering and Technology, 2004.

[26] Cohen, L., *Time-Frequency Analysis.* New Jersey: Prentice Hall PTR, 1995.

[27] Bishop, C. M., *Pattern Recognition and Machine Learning.* Springer Science + Business Media, LLC, 2006.

[28] Chiuso, A., “Regularization and Bayesian learning in Dynamical Systems: Past, Present and Future,” *Annual Reviews in Control*, vol. 41, pp. 24–38, 2016.

[29] Pillonetto, G., Dinuzzo, F., Chen, T. *et al.*, “Kernel methods in system identification, machine learning and function estimation: A survey,” *Automatica*, vol. 50, no. 3, pp. 657–682, 2014.

[30] Darwish, M. A. H., “Bayesian Identification of Linear Dynamic Systems. Synthesis of Kernels in the LTI Case and Beyond,” Ph.D. dissertation, Technische Universiteit Eindhoven, 2017.

[31] Åström, K.-J. and Torsten, B., “Numerical Identification of Linear Dynamic Systems from Normal Operating Records,” *IFAC Proceedings Volumes*, vol. 2, no. 2, pp. 96–111, 1965.

[32] Hurvich, C. M. and Tsai, C.-L., “Regression and Time Series Model Selection in Small Samples,” *Biometrika*, vol. 76, no. 2, pp. 297–307, 1989.

[33] Akaike, H., “A New Look at the Statistical Model Identification,” *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.

[34] Schwarz, G., “Estimating the Dimension of a Model,” *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[35] Leeb, H. and Pötscher, B. M., “Model Selection and Inference: Facts and Fiction,” *Econometric Theory*, vol. 21, no. 1, pp. 21–59, 2005.

[36] Katayama, T. and Tanaka, H., “An approach to closed-loop subspace identification by orthogonal decomposition,” *Automatica*, vol. 43, no. 9, pp. 1623–1630, 2007.

[37] Lin, W., Qin, S. J., and Ljung, L., “Comparisons of Subspace Identification Methods for Systems Operating on Closed-Loop,” *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 494–499, 2005.

[38] Chou, C. T. and Verhaegen, M., “Closed-Loop Identification Using Canonical Correlation Analysis,” in *European Control Conference*, 1999, pp. 311–315.

[39] Kalman, R. E., “A New Approach to Linear Filtering and Prediction Problems,” *Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[40] Grewal, M. S. and Andrews, A. P., *Kalman Filtering: Theory and Practice Using MATLAB.* John Wiley & Sons, Inc., 2001.

[41] Julier, S. J. and Uhlmann, J. K., “A New Extension of the Kalman Filter to Nonlinear Systems,” *Proceedings of SPIE*, pp. 182–193, 1999.

[42] Wan, E. A. and Van Der Merwe, R., "The Unscented Kalman Filter for Nonlinear Estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158.

[43] Helmicki, A. J., Jacobson, C. A., and Nett, C. N., "Control Oriented System Identification: A Worst-Case/Deterministic Approach in H∞," *IEEE Transactions on Automatic Control*, vol. 36, no. 10, pp. 1163–1176, 1991.

[44] Milanese, M. and Vicino, A., "Optimal estimation theory for dynamic systems with set membership uncertainty. An overview," pp. 997–1009, 1991.

[45] Casini, M., "Set-membership Estimation: an Advanced Tool for System Idenification," Ph.D. dissertation, Universita Degli Studi di Siena, 2002.

[46] Narenda, K. S. and Parthasarathy, K., "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.

[47] Dror, S., "Identification and Control of Non-Linear Time-Varying Dynamical Systems Using Artificial Neural Networks," Ph.D. dissertation, Naval Postgraduate School, 1992.

[48] Polycarpou, M. M. and Ioannou, P. A., "Neural networks as on-line approximators of nonlinear systems," in *Proceedings of the 31st IEEE Conference on Decision and Control*, 1992, pp. 7–12.

[49] Tutunji, T. A., "Parametric system identification using neural networks," *Applied Soft Computing Journal*, vol. 47, pp. 251–261, 2016.

[50] Isermann, R., Ernst, S., and Nelles, O., "Identification with Dynamic Neural Networks - Architectures, Comparisons, Applications," *IFAC Proceedings Volumes*, vol. 30, no. 11, pp. 947–972, 1997.

[51] Nelles, O., *Nonlinear System Identification. From Classical Approaches to Neural Networks and Fuzzy Models.* Springer-Verlag Berlin Heidelberg New York, 2001.

[52] Leondes, C. T., *Control and Dynamic Systems. Neural Network Systems Techniques and Applications.* Academic Press, Inc., 1998.

[53] Ogunmolu, O., Gu, X., Jiang, S., and Gans, N., "Nonlinear Systems Identification Using Deep Dynamic Neural Networks," in *American Control Conference*, 2017.

[54] de la Rosa, E. and Yu, W., "Randomized algorithms for nonlinear system identification with deep learning modification," *Information Sciences*, vol. 364-365, pp. 197–212, 2016.

[55] Bayes, T., “An essay towards solving a problem in the doctrine of chances,” *Philosophical Transactions of the Royal Society of London*, vol. 53, pp. 370–418, 1763.

[56] Wright, S., “Correlation and Causation,” *Journal of Agricultural Research*, vol. 20, pp. 557–585, 1921.

[57] Wright, S., “The Method of Path Coefficients,” *Annals of Mathematical Statistics*, vol. 5, pp. 161–215, 1934.

[58] Wold, H., “Causality and Econometrics,” *Econometrica*, vol. 22, no. 2, pp. 162–177, 1954.

[59] Blalock Jr., H. M., *Causal Models in the Social Sciences.* Chicago, Illinois: Aldine-Atheson, 1971.

[60] Elston, R. and Stewart, J., “A general model for the genetic analysis of pedigree data.” *Human heredity*, vol. 21, no. 6, pp. 523–542, 1971.

[61] Lange, K. and Elston, R., “Extensions to Pedigree Analysis. I. Likelihood Calculations for Simple and Complex Pedigrees,” *Human Heredity*, vol. 25, pp. 95–105, 1975.

[62] Cox, D. and Hinkley, D., *Theoretical Statistics.* London: Chapman and Hall, 1974.

[63] Feller, W., *An Introduction to Probability Theory and Its Applications (third edition).* New York: John Wiley & Sons, 1970, vol. I.

[64] Gelman, A., “Objections to Bayesian statistics,” *Bayesian Analysis*, vol. 3, no. 3, pp. 445–450, 2008.

[65] Jaynes, E. T., *Probability Theory: The Logic of Science.* New York: Cambridge University Press, 2003.

[66] Dienes, Z., “Bayesian versus orthodox statistics: Which side are you on?” *Perspectives on Psychological Science*, vol. 6, no. 3, pp. 274–290, 2011.

[67] Ledley, R. S. and Lusted, L. B., “Reasoning foundations of medical diagnosis,” *Science*, vol. 130, no. 3366, pp. 9–21, 1959.

[68] Ledley, R. S. and Lusted, L. B., “The Use of Electronic Computers to Aid in Medical Diagnosis,” *Proceedings of the IRE*, vol. 47, no. 11, pp. 1970–1977, 1959.

[69] Warner, H. R., Toronto, A. F., Veasy, L. G., and Stephenson, R., “A mathematical approach to medical diagnosis: Application to congenital heart disease,” *Journal of the American Medical Association*, vol. 177, pp. 177–183, 1961.

[70] Horrocks, J. C., McCann, A. P., Staniland, J. R. *et al.*, "Computer-aided Diagnosis : Description of an Adaptable System, and Operational Experience with 2,034 Cases," *British Medical Journal*, vol. 2, no. 5804, pp. 5–9, 1972.

[71] de Dombal, F. . T., Leaper, D. J., Staniland, J. R. *et al.*, "Computer-aided diagnosis of acute abdominal pain," *British Medical Journal*, vol. 2, no. 5804, pp. 9–13, 1972.

[72] de Dombal, F. . T., Horrocks, J. C., Staniland, J. R., and Guillou, P. J., "Production of Artificial "Case Histories" by Using a Smal Computer," *British Medical Journal*, vol. 2, no. 5761, pp. 578–581, 1971.

[73] Leaper, D. J., Horrocks, J. C., Staniland, J. R., and de Dombal, F. . T., "Computer-Assisted Diagnosis of Abdominal Pain using "Estimates" Provided by Clinicians," *British Medical Journal*, vol. 4, no. 5836, pp. 350–354, 1972.

[74] Wermuth, N. and Lauritzen, S. L. ., "Graphical and Recursive Models for Contingency Tables," *Biometrica*, vol. 70, no. 3, pp. 537–552, 1983.

[75] Cooper, G. F., "Nestor: A Computer-Based Medical Diagnostic Aid that Integrates Causal and Probabilistic Knowledge," Ph.D. dissertation, Stanford University, 1984.

[76] Andreassen, S., Woldbye, M., Falck, B., and Andersen, S. K., "MUNIN: A causal probabilistic network for interpretation of electromyographic findings," in *Proceedings of the 10th international joint conference on Artificial intelligence-Volume 1*, 1987, pp. 366–372.

[77] Pearl, J., "Fusion, Propogation and Structuring in Belief Networks," *Artificial Inteligence*, vol. 29, no. 3, pp. 241–288, 1986.

[78] Lauritzen, S. L. and Spiegelhalter, D. J., "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 50, no. 2, pp. 157–224, 1988.

[79] Howard, R. A. and Matheson, J. E., "Influence Diagrams," *Decision Analysis*, vol. 2, no. 3, pp. 127–143, 2005.

[80] Howard, R. A., "Knowledge Maps," *Management Science*, vol. 35, no. 8, pp. 903–922, 1989.

[81] "A. M. Turing award laureats - Judea Pearl," [cited on 31.10.2019]. [Online]. Available: https://amturing.acm.org/award_winners/pearl_2658896.cfm

[82] Pearl, J., "Bayesian Networks A model of self-activated memory for evidential reasoning.pdf," in *the Seventh Annual Conference of the Cognitive Science Society*, 1985, pp. 329–334.

[83] Kim, J. H. and Pearl, J., “A Computational Model for Causal and Diagnostic Reasoning in Inference Systems,” in *Proceedings of the Eighth international joint conference on Artificial intelligence*, 1983, pp. 190–193.

[84] Pearl, J., “A Constraint - Propagation Approach to Probabilistic Reasoning,” in *Proceedings of Workshop on Uncertainty and Probability in Artificial Inteligence*, 1985, pp. 357–369.

[85] Pearl, J. and Verma, T., “The Logic of Representing Dependencies by Directed Graphs,” in *Proceedings of the Sixth National Conference on Artificial Inteligence*, 1987, pp. 374–379.

[86] “Judea Pearl’s homepage,” [cited on 31.10.2019]. [Online]. Available: http://bayes.cs.ucla.edu/jp_home.html

[87] Pearl, J., *Probabilistic Reasoning in Intelligent Systems.* San Mateo: Morgan Kaufmann, 1988.

[88] Shachter, R. D., “Evaluating Influence Diagrams,” *Operations Research*, vol. 34, no. 6, pp. 871–882, 1986.

[89] Shachter, R. D. and Heckerman, D. E., “Thinking backward for knowledge acquisition,” *AI Magazine*, vol. 8, no. 3, pp. 55–61, 1987.

[90] Smith, J. Q., “Influence Diagrams for Statistical Modelling,” *The Annals of Statistics*, vol. 17, no. 2, pp. 654–672, 1989.

[91] Neapolitan, R., *Probabilistic Reasoning in Expert Systems: Theory and Algorithms.* New York: J. Wiley, 1989.

[92] Andreassen, S., Jensen, F. V., Andersen, S. K. *et al.*, “MUNIN - an Expert EMG Assistant,” in *Computer-Aided Electromyography and Expert Systems.* Elsevier (Noth-Holland), 1989, ch. 21.

[93] Shwe, M. A., Middleton, B., Heckerman, D. E. *et al.*, “Probabilistic Diagnosis Using a Reformulation of the INTERNIST-1/QMR Knowledge Base. I. The Probabilistic Model and Inference Algorithms,” *Methods of Information in Medicine*, vol. 30, pp. 241–255, 1991.

[94] Middleton, B., Shwe, M. A., Heckerman, D. E. *et al.*, “Probabilistic Diagnosis Using a Reformulation of the Internist-1/ QMR Knowledge Base II. Evaluation of Diagnostic Performance,” *Methods of Information in Medicine*, vol. 30, pp. 256–267, 1991.

[95] Heckerman, D. E., Horvitz, E. J., and Nathwani, B. N., “Toward normative expert systems: Part I. The Pathfinder project,” *Methods of Information in Medicine*, vol. 31, pp. 90–105, 1992.

[96] Heckerman, D. E. and Nathwani, B. N., “An evaluation of the diagnostic accuracy of Pathfinder,” *Computers and Biomedical Research*, vol. 25, pp. 56–74, 1992.

[97] Heckerman, D. E., “Toward Normative Expert Systems : Part II Probability-Based Representations for Efficient Knowledge Acquisition and Inference,” *Methods of Information in Medicine*, vol. 31, pp. 106–116, 1992.

[98] Horvitz, E., Ruokangas, C., Srinivas, S., and Barry, M., “A Decision-Theoretic Approach to the Display of Information for Time-Critical Decisions : The Vista Project,” in *Proceedings of the Conference on Space Operations and Automation and Research*, 1992.

[99] Horvitz, E. and Barry, M., “Display of Information for Time-Critical Decision Making,” in *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, 1995, pp. 296–305.

[100] Horvitz, E., Breese, J., Heckerman, D. *et al.*, “The Lumiere Project : Bayesian User Modeling for Inferring the Goals and Needs of Software Users,” in *Proceedings of the Fourteens Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 256–265.

[101] Pourret, O., Naim, P., and Marcot, B., Eds., *Bayesian Networks. A Practical Guide to Applications.* John Wiley & Sons, 2008.

[102] Neapolitan, R., *Probabilistic Methods for Bioinformatics: with an Introduction to Bayesian Networks.* Morgan Kaufmann, 2009.

[103] Friedman, N., Linial, M., Nachman, I., and Pe’er, D., “Using Bayesian Networks to Analyze Expression Data,” in *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology*, 2000, pp. 127–135.

[104] Jiang, X., Neapolitan, R. E., Barmada, M. M., and Visweswaran, S., “Learning genetic epistasis using Bayesian network scoring criteria,” *BMC Bioinformatics*, pp. 1–12, 2011.

[105] Yan, L. J. and Cercone, N., “Bayesian network modeling for evolutionary genetic structures,” *Computers and Mathematics with Applications*, vol. 59, no. 8, pp. 2541–2551, 2010.

[106] Njah, H. and Jamoussi, S., “Weighted ensemble learning of Bayesian network for gene regulatory networks,” *Neurocomputing*, vol. 150, pp. 404–416, 2015.

[107] Allen, D. and Darwiche, A., “RC _ Link : Genetic linkage analysis using Bayesian networks,” *International Journal of Approximate Reasoning*, vol. 48, pp. 499–525, 2008.

[108] Constantinou, A. C., Fenton, N., Marsh, W., and Radlinski, L., "From complex questionnaire and interviewing data to intelligent Bayesian network models for medical decision support," *Artificial Intelligence In Medicine*, vol. 67, pp. 75–93, 2016.

[109] Mani, S., Valtorta, M., and McDermott, S., "Building Bayesian Network Models in Medicine : The MENTOR Experience," *Applied Intelligence*, vol. 22, pp. 93–108, 2005.

[110] Nikovski, D., "Constructing Bayesian Networks for Medical Diagnosis from Incomplete and Partially Correct Statistics," *IEEE Transactions on Knowledge and Data Engineering*, vol. 12, no. 4, pp. 509–516, 2000.

[111] Montironi, R., Whimster, W. F., Collan, Y. *et al.*, "How to develop and use a Bayesian Belief Network," *J Clin Pathol*, vol. 49, no. 3, pp. 194–201, 1996.

[112] Vila-francés, J., Sanchís, J., Soria-Olivas, E. *et al.*, "Expert System for Predicting Unstable Angina on Bayesian Networks," *Expert Systems with Applic ations journal*, vol. 40, pp. 5004–5010, 2013.

[113] Seixas, L. F., Zadrozny, B., Laks, J. *et al.*, "A Bayesian network decision model for supporting the diagnosis of dementia , Alzheimer' s disease and mild cognitive impairment," *Computers in Biology and Medicine*, vol. 51, pp. 140–158, 2014.

[114] Kuang, D., Yang, R., Chen, X. *et al.*, "Depression recognition according to heart rate variability using Bayesian Networks," *Journal of Psychiatric Research*, vol. 95, pp. 282–287, 2017.

[115] Jagdale, S., Kolekara, M. H., and Khot, U. P., "Smart Sensing using Bayesian Network for Computer Aided Diagnostic Systems," *Procedia Computer Science*, vol. 45, pp. 762–769, 2015.

[116] Antal, P., Fannes, G., Timmerman, D. *et al.*, "Using Literature and Data to Learn Bayesian Networks as Clinical Models of Ovarian Tumors," *Artificial Intelligence In Medicine*, vol. 30, no. 3, pp. 257–281, 2004.

[117] Alvarez, S. M., Poelstra, B. A., and Burd, R. S., "Evaluation of a Bayesian decision network for diagnosing pyloric stenosis," *Journal of Pediatric Surgery*, vol. 41, pp. 155–161, 2006.

[118] Cruz-Ramírez, N., Acosta-Mesa, H. G., Carrillo-Calvet, H. *et al.*, "Diagnosis of breast cancer using Bayesian networks : A case study," *Computers in Biology and Medicine*, vol. 37, pp. 1553–1564, 2007.

[119] Luciani, D., Cavuto, S., Antiga, L. *et al.*, "Bayes Pulmonary Embolism Assisted Diagnosis: a New Expert System for Clinical Use," *Emergency Medicine Journal*, vol. 24, no. 3, pp. 157–164, 2007.

[120] Burnside, E. S., Rubin, D. L., Fine, J. P. *et al.*, "Bayesian Network to Predict Breast Cancer Risk of Mammographic Microcalcifications and Reduce Number of Benign Biopsy Results : Initial Experience 1," *Radiology*, vol. 240, no. 3, pp. 666–673, 2006.

[121] Uebersax, J. S., "Breast Cancer Risk Modeling : An Application of Bayes Nets," RavenPack International, Tech. Rep., 2004.

[122] Sanders, D. L. and Aronsky, D., "Detecting Asthma Exacerbations in a Pediatric Emergency Department Using a Bayesian Network," in *AMIA Symposium Proceedings*, 2006, pp. 684–688.

[123] Petousis, P., Han, S. X., Aberle, D., and Bui, A. A. T., "Prediction of lung cancer incidence on the low-dose computed tomography arm of the National Lung Screening Trial : A dynamic Bayesian network," *Artificial Intelligence In Medicine*, vol. 72, pp. 42–55, 2016.

[124] Haddawy, P., Hasan, A. H. M. I., Kasantikul, R. *et al.*, "Spatiotemporal Bayesian networks for malaria prediction," *Artificial Intelligence In Medicine*, vol. 84, pp. 127–138, 2018.

[125] Kline, J. A., Novobilski, A. J., Kabrhel, C. *et al.*, "Derivation and Validation of a Bayesian Network to Predict Pretest Probability of Venous Thromboembolism," *Annals of Emergency Medicine*, vol. 45, no. 3, pp. 282–290, 2005.

[126] Verduijn, M., Peek, N., Rosseel, P. M. J. *et al.*, "Prognostic Bayesian networks I : Rationale , learning procedure , and clinical use," *Journal of Biomedical Informatics*, vol. 40, pp. 609–618, 2007.

[127] Verduijn, M., Rosseel, P. M. J., Peek, N. *et al.*, "Prognostic Bayesian networks II : An application in the domain of cardiac surgery," *Journal of Biomedical Informatics*, vol. 40, pp. 619–630, 2007.

[128] Corani, G., Magli, C., Giusti, A. *et al.*, "A Bayesian network model for predicting pregnancy after in vitro fertilization," *Computers in Biology and Medicine*, vol. 43, pp. 1783–1792, 2013.

[129] Hoot, N. and Aronsky, D., "Using Bayesian Networks to Predict Survival of Liver Transplant Patients," in *AMIA Symposium Proceedings*, 2005, pp. 345–349.

[130] Sandri, M., Berchialla, P., Baldi, I. *et al.*, "Dynamic Bayesian Networks to predict sequences of organ failures in patients admitted to ICU," *Journal of Biomedical Informatics*, vol. 48, pp. 106–113, 2014.

[131] Hernandez-Leal, P., Rios-Flores, A., Ávila-Rios, S. *et al.*, "Discovering human immunodeficiency virus mutational pathways using temporal Bayesian networks," *Artificial Intelligence In Medicine*, vol. 57, pp. 185–195, 2013.

[132] Mumford, J. A. and Ramsey, J. D., “Bayesian networks for fMRI : A primer,” *NeuroImage*, vol. 86, pp. 573–582, 2014.

[133] Ide, J. S., Zhang, S., and Li, C.-s. R., “Bayesian network models in brain functional connectivity analysis,” *International Journal of Approximate Reasoning*, vol. 55, pp. 23–35, 2014.

[134] Fenton, N. and Neil, M., “Comparing risks of alternative medical diagnosis using Bayesian arguments,” *Journal of Biomedical Informatics*, vol. 43, pp. 485–495, 2010.

[135] Aitken, C., Garbolino, P., Bozza, S. *et al.*, *Bayesian Networks for Probabilistic Inference and Decision Analysis in Forensic Science, 2nd Edition.* John Wiley & Sons, 2014.

[136] Dawid, A. P., Mortera, J., and Vicard, P., “Representing and solving complex DNA identification cases using Bayesian networks,” *International Congress Series*, vol. 1288, pp. 484–491, 2006.

[137] Constantinou, A. C., Freestone, M., Marsh, W. *et al.*, “Risk assessment and risk management of violent reoffending among prisoners,” *Expert Systems with Applications*, vol. 42, pp. 7511–7529, 2015.

[138] Constantinou, A. C., Freestone, M., Marsh, W., and Coid, J., “Causal inference for violence risk management and decision support in forensic psychiatry,” *Decision Support Systems*, vol. 80, pp. 42–55, 2015.

[139] De Zoete, J., Sjerps, M., Lagnado, D., and Fenton, N., “Modelling crime linkage with Bayesian networks,” *Science and Justice*, vol. 55, pp. 209–217, 2015.

[140] Aguilera, P. A., Fernández, A., Fernández, R. *et al.*, “Bayesian networks in environmental modelling,” *Environmental Modelling and Software*, vol. 26, pp. 1376–1388, 2011.

[141] Lohr, C., Wenger, A., Woodberry, O. *et al.*, “Predicting island biosecurity risk from introduced fauna using Bayesian Belief Networks,” *Science of the Total Environment*, vol. 601-602, pp. 1173–1181, 2017.

[142] Uusitalo, L., Tomczak, M. T., Müller-Karulis, B. *et al.*, “Hidden variables in a Dynamic Bayesian Network identify ecosystem level change,” *Ecological Informatics*, vol. 45, pp. 9–15, 2018.

[143] Franco, C., Appleby Hepburn, L., Smith, D. J. *et al.*, “A Bayesian Belief Network to assess rate of changes in coral reef ecosystems,” *Environmental Modelling and Software*, vol. 80, pp. 132–142, 2016.

[144] Jäger, W. S., Christie, E. K., Hanea, A. M. *et al.*, “A Bayesian network approach for coastal risk analysis and decision making,” *Coastal Engineering*, vol. 134, pp. 48–61, 2018.

[145] Borunda, M., Jaramillo, O. A., Reyes, A., and Ibargüengoytia, P. H., “Bayesian networks in renewable energy systems : A bibliographical survey,” *Renewable and Sustainable Energy Reviews*, vol. 62, pp. 32–45, 2016.

[146] Drury, B., Valverde-Rebaza, J., Moura, M.-F., and de Andrade Lopes, A., “A survey of the applications of Bayesian networks in agriculture,” *Engineering Applications of Artificial Intelligence*, vol. 65, pp. 29–42, 2017.

[147] del Sagrado, J., Sánchez, J. A., Rodríguez, F., and Berenguel, M., “Bayesian networks for greenhouse temperature control,” *Journal of Applied Logic*, vol. 17, pp. 25–35, 2016.

[148] Taalab, K., Corstanje, R., Zawadzka, J. *et al.*, “On the application of Bayesian Networks in Digital Soil Mapping,” *Geoderma*, vol. 259-260, pp. 134–148, 2015.

[149] Zhu, J., Ge, Z., Song, Z. *et al.*, “Large-scale plant-wide process modeling and hierarchical monitoring : A distributed Bayesian network approach,” *Journal of Process Control*, vol. 65, pp. 91–106, 2018.

[150] Carbery, C. M., Woods, R., and Marshall, A. H., “A Bayesian network based learning system for modelling faults in large-scale manufacturing,” in *IEEE International Conference on Industrial Technology (ICIT)*, 2018, pp. 1357–1362.

[151] Verron, S., Tiplica, T., and Kobi, A., “Fault diagnosis of industrial systems by conditional Gaussian network including a distance rejection criterion,” *Engineering Applications of Artificial Intelligence*, vol. 23, pp. 1229–1235, 2010.

[152] Verron, S., Li, J., and Tiplica, T., “Fault detection and isolation of faults in a multivariate process with Bayesian network,” *Journal of Process Control*, vol. 20, pp. 902–911, 2010.

[153] Jun, H.-B. and Kim, D., “A Bayesian network-based approach for fault analysis,” *Expert Systems With Applications*, vol. 81, pp. 332–348, 2017.

[154] Barua, S., Gao, X., Pasman, H., and Mannan, M. S., “Bayesian network based dynamic operational risk assessment,” *Journal of Loss Prevention in the Process Industries*, vol. 41, pp. 399–410, 2016.

[155] Liu, Q., Pérès, F., and Tchangani, A., “Object Oriented Bayesian Network for Complex System Risk Assessment,” *IFAC-PapersOnLine*, vol. 49, no. 28, pp. 31–36, 2016.

[156] Liu, Q., Tchangani, A., and Pérès, F., “Modelling complex large scale systems using object oriented Bayesian networks (OOBN),” *IFAC-PapersOnLine*, vol. 49, no. 12, pp. 127–132, 2016.

[157] Argenti, F., Landucci, G., Reniers, G., and Cozzani, V., “Vulnerability assessment of chemical facilities to intentional attacks based on Bayesian Network,” *Reliability Engineering and System Safety*, vol. 169, pp. 515–530, 2018.

[158] Tang, C., Yi, Y., Yang, Z., and Sun, J., "Risk analysis of emergent water pollution accidents based on a Bayesian Network," *Journal of Environmental Management*, vol. 165, pp. 199–205, 2016.

[159] Sýkora, M., Marková, J., and Diamantidis, D., "Bayesian network application for the risk assessment of existing energy production units," *Reliability Engineering and System Safety*, vol. 169, pp. 312–320, 2018.

[160] Wang, Y. F., Qin, T., Li, B. *et al.*, "Fire probability prediction of offshore platform based on Dynamic Bayesian Network," *Ocean Engineering*, vol. 145, pp. 112–123, 2017.

[161] Averill, A. F., Ingram, J. M., Holborn, P. G. *et al.*, "Application of Bayesian methods and networks to ignition hazard event prediction in nuclear waste decommissioning operations," *Process Safety and Environmental Protection*, vol. 116, pp. 396–404, 2018.

[162] Codetta-Raiteri, D., "Applying Generalized Continuous Time Bayesian Networks to a Reliability Case Study," *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 676–681, 2015.

[163] Weber, P. and Jouffe, L., "Reliability modelling with dynamic bayesian networks," *IFAC Proceedings*, vol. 36, no. 5, pp. 57–62, 2003.

[164] Wilson, A. G. and Huzurbazar, A. V., "Bayesian networks for multilevel system reliability," *Reliability Engineering and System Safety*, vol. 92, pp. 1413–1420, 2007.

[165] Neapolitan, R. E. and Jiang, X., *Probabilistic Methods for Financial and Marketing Informatics*. San Francisco: Morgan Kaufmann, 2007.

[166] Constantinou, A. C., "Bayesian networks for prediction, risk assessment and decision making in an inefficient Association Football gambling market," Ph.D. dissertation, Queen Mary University of London, 2012.

[167] Almond, R., Mislevy, R., Steinberg, L. *et al.*, *Bayesian Networks in Educational Assessment*. New York: Springer-Verlag, 2015.

[168] Almond, R. G., Shute, V. J., Underwood, J. S., and Zapata-Rivera, J.-D., "Bayesian networks : A teacher's view," *International Journal of Approximate Reasoning*, vol. 50, pp. 450–460, 2009.

[169] Das, M. and Ghosh, S. K., "semBnet: A semantic Bayesian network for multivariate prediction of meteorological time series data," *Pattern Recognition Letters*, vol. 93, pp. 192–201, 2017.

[170] de Campos, L. M., Fernández-Luna, J. M., and Huete, J. F., "Bayesian networks and information retrieval : an introduction to the special issue," *Information Processing and Management*, vol. 40, pp. 727–733, 2004.

[171] Al-garadi, M. A., Khan, M. S., Varathan, K. D. *et al.*, "Using online social networks to track a pandemic : A systematic review," *Journal of Biomedical Informatics*, vol. 62, pp. 1–11, 2016.

[172] Varshney, D., Kumar, S., and Gupta, V., "Predicting information diffusion probabilities in social networks : A Bayesian networks based approach," *Knowledge-Based Systems*, vol. 133, pp. 66–76, 2017.

[173] Dabrowski, J. J. and de Villiers, J. P., "Maritime piracy situation modelling with dynamic Bayesian networks," *Information Fusion*, vol. 23, pp. 116–130, 2015.

[174] Freire, A., Perkusich, M., Saraiva, R. *et al.*, "A Bayesian networks-based approach to assess and improve the teamwork quality of agile teams," *Information and Software Technology*, vol. 100, pp. 119–132, 2018.

[175] Chen, G., Li, G., Nie, Y. *et al.*, "Stylistic indoor colour design via Bayesian network," *Computers and Graphics*, vol. 60, pp. 34–45, 2016.

[176] Shachter, R. D. and Kenley, C. R., "Gaussian Influence Diagrams," *Management Science*, vol. 35, no. 5, pp. 527–550, 1989.

[177] Poland, W. B. I., "Decision Analysis with Continuous and Discrete Variables: A Mixture Distribution Approach," Ph.D. dissertation, Stanford University, 1994.

[178] Dean, T. and Kanazawa, K., "A Model for Reasoning About Persistence and Causation," *Computational Intelligence*, vol. 5, no. 3, pp. 142–150, 1989.

[179] Murphy, K. P., "The Bayes Net Toolbox for Matlab," *Computing Science and Statistics*, vol. 33, no. 2, pp. 1024–1034, 2001.

[180] "Bayes Net Toolbox," [cited on 31.10.2019]. [Online]. Available: https://github.com/bayesnet/bnt

[181] "MATLAB software," [cited on 31.10.2019]. [Online]. Available: https://www.mathworks.com/products/matlab.html

[182] "Software Packages for Graphical Models," [cited on 31.10.2019]. [Online]. Available: https://www.cs.ubc.ca/~murphyk/Software/bnsoft.html

[183] Herzallah, R. and Lowe, D., "A Bayesian Perspective on Stochastic Neuro control," *IEEE Transactions on Neural Networks*, vol. 19, pp. 914–924, 2008.

[184] Tagaras, G. and Nikolaidis, Y., "Comparing the Effectiveness of Various Bayesian x Control Charts," *Operations Research*, 2002.

[185] Herzallah, R. and Lowe, D., "A Bayesian Approach to Modeling the Conditional Density of the Inverse Controller," in *Proceedings of the 2004 IEEE International Conference on Control Applications*, 2004, pp. 788–793.

[186] Oakland John S., *Statistical Process Control. Fifth Edition.* Oxford: Butterworth-Heinemann, 2003.

[187] Girshick, M. A. and Rubin, H., "A Bayes Approach to a Quality Control Model," *The Annals of Mathematical Statistics*, vol. 1, pp. 114–125, 1952.

[188] Nikolaidis, Y. and Tagaras, G., "New indices for the evaluation of the statistical properties of Bayesian x control charts for short runs," *European Journal of Operational Research*, vol. 259, pp. 280–292, 2017.

[189] Naderkhani, F. and Makis, V., "Economic design of multivariate Bayesian control chart with two sampling intervals," *International Journal of Production Economics*, vol. 174, pp. 29–42, 2016.

[190] Peterka, V., "Bayesian Approach to System Identification," in *Trends and Progress in System Identification*, Pieter Eykhoff, Ed. Pergamon, 1981, ch. 8, pp. 239–304.

[191] Mosterman, P. J. and Biswas, G., "Monitoring, Prediction, and Fault Isolation in Dynamic Physical Systems," in *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Applications of Artificial Intelligence Conference*, 1997, pp. 100–105.

[192] Lerner, U. N., "Hybrid Bayesian Networks for Reasoning about Complex Systems," Ph.D. dissertation, Stanford University, 2002.

[193] Ould Bouamama, B., Biswas, G., Loureiro, R., and Merzouki, R., "Graphical methods for diagnosis of dynamic systems: Review," *Annual Reviews in Control*, vol. 38, no. 2, pp. 199–219, 2014.

[194] Mori, J. and Yu, J., "Dynamic Bayesian network based networked process monitoring for fault propagation identification and root cause diagnosis of complex dynamic processes," in *10th IFAC International Symposium on Dynamics and Control of Process Systems.* IFAC, 2013, pp. 678–683.

[195] Rabiner, L. R. and Juang, B. H., "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[196] Rabiner, L. R., "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[197] Heck, L. P. and McClellan, J. H., "Mechanical System Monitoring Using Hidden Markov Models," in *Acoustics, Speech, and Signal Processing*, 1991, pp. 1697–1700.

[198] Smyth, P., "Hidden Markov Models for Fault Detection in Dynamic Systems," *Pattern Recognition*, vol. 27, no. 1, pp. 149–164, 1994.

[199] Fraser, A. M., *Hidden Markov Models and Dynamical Systems.* Philadelphia: Society for Industrial and Applied Mathematics, 2008.

[200] Fine, S., Singer, Y., and Tishby, N., "The Hierarchical Hidden Markov Model : Analysis and Applications," *Machine Learning*, vol. 32, pp. 41–62, 1998.

[201] Frankel, J., "Linear dynamic models for automatic speech recognition," Ph.D. dissertation, University of Edinburgh, 2003.

[202] Minka, T. P., "From Hidden Markov Models to Linear Dynamical Systems," pp. 1–10, 1999.

[203] Ghahramani, Z. and Hinton, G. E., "Parameter Estimation for Linear Dynamical Systems," pp. 1–6, 1996.

[204] Ghahramani, Z., "An Introduction to Hidden Markov Models and Bayesian Networks," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 1, pp. 9–42, 2001.

[205] Jordan, M. I., Ed., *Learning in Graphical Models.* Cambridge MA: The MIT Press, 1998.

[206] Pavlović, V., Rehg, J. M., Cham, T.-J., and Murphy, K. P., "A Dynamic Bayesian Network Approach to Figure Tracking Using Learned Dynamic Models," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999.

[207] Vidal, R., Chiuso, A., and Soatto, S., "Observability and Identifiability of Jump Linear Systems," in *Proceedings of the 41st IEEE Conference on Decision and Control*, 2002, pp. 3614–3619.

[208] Ashcroft, M., "An introduction to Bayesian networks in systems and control," *18th International Conference on Automation and Computing (ICAC)*, pp. 1–6, 2012.

[209] Spirtes, P., "Directed Cyclic Graphical Representations of Feedback Models," in *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, 1995, pp. 491–498.

[210] Heckerman, D., Chickering, D. M., Meek, C. *et al.*, "Dependency Networks for Inference, Collaborative Filtering, and Data Visualization," *Journal of Machine Learning Research*, vol. 1, pp. 49–75, 2000.

[211] Deventer, R., Denzler, J., and Niemann, H., "Bayesian Controller Versus Traditional Controllers," *International Conference on Computational Intelligence for Modelling Control and Automation*, 2003.

[212] Deventer, R., Denzler, J., and Niemann, H., "Using Non-Markov models for the Control of Dynamic Systems," 2003.

[213] Deventer, R., Denzler, J., and Niemann, H., "Non-linear Modelling of a Production Process by Hybrid Bayesian Networks," in *Proceedings of the 14th European Conference on Artificial Intelligence*, 2000, pp. 576–580.

[214] Sethi, S. P. and Zhang, Q., *Systems & Control : Foundations & Applications.* Birkhausser Boston, 1994.

[215] Hommersom, A. and Lucas, P. J., "Using Bayesian Networks in an Industrial Setting: Making Printing Systems Adaptive," in *Proceedings of the 19th European Conference on Artificial Intelligence*, 2010, pp. 401–406.

[216] Evers, S. and Lucas, P. J., "Constructing Bayesian Networks for Linear Dynamic Systems," in *Bayesian Modeling Applications Workshop*, 2011, pp. 26–33.

[217] Toussaint, M. and Goerick, C., "A Bayesian view on motor control and planning," in *From Motor Learning to Interaction Learning in Robots*, Sigaud, O. and Peters, J., Eds. Berlin: Springer, 2010, p. 538.

[218] Lee, S.-H. and Song, J., "Bayesian-network-based system identification of spatial distribution of structural parameters," *Engineering Structures*, vol. 127, pp. 260–277, 2016.

[219] Tran, T. B., "A Bayesian Network Framework for Probabilistic Identification of Model Parameters from Normal and Accelerated tests : Application to Chloride Ingress into Conrete," Ph.D. dissertation, Université de Nantes, 2015.

[220] Richard, B., Adelaide, L., and Cremona, C., "A Bayesian approach to estimate material properties from global statistical data," *European Journal of Environmental and Civil Engineering*, pp. 1–11, 2012.

[221] Brogan, W. L., *Modern Control theory. Third edition.* New Jersey: Prentice-Hall, Inc., 1991.

[222] Paoletti, S., Juloski, A. L., Ferrari-Trecate, G., and Vidal, R., "Identification of Hybrid Systems. A Tutorial," *European Journal of Control*, vol. 13, pp. 242–260, 2007.

[223] Ali, J. M., Hoang, N. H., Hussain, M. A., and Dochain, D., "Review and Classification of Recent Observers Applied in Chemical Process Systems," *Computers and Chemical Engineering*, vol. 76, pp. 27–41, 2015.

[224] Jensen, F. V. and Nielsen, T. D., *Bayesian Networks and Decision Graphs. Second Edition.* Science + Business Media, LLC, 2007.

[225] Conrady, S. and Jouffe, L., *Bayesian Networks & Bayesia Lab - A Practical Inroduction for Researchers.* Bayesia USA, 2015.

[226] "Hugin Lite software," [cited on 31.10.2019]. [Online]. Available: https://www.hugin.com/index.php/hugin-lite/

[227] Shannon, C. E., "Communication in the Presence of Noise," *Proceedings of the Institute of Radio Engineers*, vol. 37, no. 1, pp. 10–21, 1949.

[228] Nodelman, U., Shelton, C. R., and Koller, D., "Continuous Time Bayesian Networks," in *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, 2002, pp. 378–387.

[229] Nodelman, U., Shelton, C. R., and Koller, D., "Learning Continuous Time Bayesian Networks," in *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, 2003, pp. 1–8.

[230] Liu, M., Hommersom, A., van der Heijden, M., and Lucas, P. J., "Hybrid time Bayesian networks," *International Journal of Approximate Reasoning*, vol. 80, pp. 460–474, 2017.

[231] Ng, B. M., "Adaptive dynamic Bayesian networks," Lawrence Livermore National Library, Salt Lake City, UT, United States, Tech. Rep., 2007.

[232] Wang, Z., Kuruoglu, E. E., Yang, X. *et al.*, "Time Varying Dynamic Bayesian Network for Non-Stationary Events Modeling and Online Inference," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1553–1568, 2011.

[233] Doshi-Velez, F., Wingate, D., Tenenbaum, J., and Roy, N., "Infinite Dynamic Bayesian Networks," *Proceedings of the 28th International Conference on Machine Learning*, pp. 913–920, 2011.

[234] Boyen, X. and Koller, D., "Tractable Inference for Complex Stochastic Processes," in *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 33–42.

[235] Mutambara, A. G., *Decentralized Estimation and Control for Multisensor Systems.* CRC Press LLC, 1998.

[236] Frogner, C. and Pfeffer, A., "Discovering Weakly-Interacting Factors in a Complex Stochastic Process," in *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, 2007, pp. 1–8.

[237] Ng, B. M., Peshkin, L., and Pfeffer, A., "Factored Particles for Scalable Monitoring," in *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, 2002, pp. 1–8.

[238] Roychoudhury, I., Biswas, G., and Koutsoukos, X., "Tracking of Dynamic Physical Systems with Dynamic Bayesian Net Models Factored using Structural Observability," in *Conference on Uncertainty in Artificail Intelligence*, 2011, pp. 1–9.

Curriculum vitae

Personal data

Name:	Maryna Garan
Date of birth:	21. 08. 1987
E-mail:	maryna.garan@tul.cz
vedidk:	8846324
ORCID iD:	0000-0002-5592-5196
h-index:	2 (Scopus); 1 (WoS)

Education

2013 – now:	Doctoral study Manufacturing systems and processes TUL (Liberec, Czech republic)
2009 – 2013:	Master study (diploma with honors) Automated control systems in mechanical engineering TUL (Liberec, Czech republic)
2004 – 2009:	Master study (diploma with honors) Computer-integrated technological processes and manufactures, KNUTD (Kyjiv, Ukraine)

Work experience

5.2017 – now:	Teaching assistant TUL (Liberec, Czech republic)
3.2017 – 4.2017:	Doctoral traineeship LARIS, Université d'Angers (Angers, Francie)
3.2016 – 5.2016:	Doctoral traineeship LARIS, Université d'Angers (Angers, Francie)
8.2013 – 4.2017:	Technician TUL (Liberec, Czech republic)

Languages

Czech:	C1
English:	C1
Ukrainian:	C2
French:	B1
German:	A1
Russian:	C2

Publications

Significant publications:

[A1] Garan, M., Verron, S., Kovalenko, I., Modrlák, O., a Lepšík, P., "Parameter Estimation in Linear Dynamic Systems using Bayesian networks," in: *Proceedings of the 2019 22nd International Conference on Process Control*, Štrbské Pleso, 11 – 14. 06. 2019, IEEE, pp. 203-208, ISBN: 978-1-72813-758-2, 2019

[A2] Kovalenko, I., Ramachandran, Y., a Garan, M., "Experimental shrinkage study of ceramic DLP 3D printed parts after firing green bodies in a kiln," *MM Science Journal,* vol. 1, pp. 2767-2771, ISSN 1803-1269, 2019, `http://doi.org/10.17973/MMSJ.2019_03_2018113`

[A3] Kovalenko, I., Verron, S., Garan, M., Šafka, J., a Moučka, M., "Implementation of Non-Destructive Evaluationand Process Monitoring in DLP-based Additive Manufacturing," *Open Engineering,* vol. 7, pp. 100-105, ISSN 2391-5439, 2017, `https://doi.org/10.1515/eng-2017-0016`

[A4] Garan, M., a Kovalenko, I., "Recalculation of initial conditions for the observable canonical form of state-space representation," in: *Proceedings of the 5th International Conference on Mechatronics and Control Engineering,* Venice, 14. – 17. 12. 2016, New York: ACM, pp. 131-135, ISBN: 978-1-4503-5215-4, 2016

[A5] Kovalenko, I., a Garan, M., "Effect of UV radiation by projectors on 3D rinting," in: *Proceedings of 2016 International Conference on Biomaterials, Nanomaterials and Composite Materials,* Chengdu, 4. – 6. 11. 2016, Francie: EDP Science, 2017, `https://doi.org/10.1051/matecconf/20178801001`

[A6] Kovalenko, I., Garan, M., Shynkarenko, A., Zelený, P., a Šafka, J., "Examining the Relationship between Forces During Stereolithography 3D Printing and Geometric Parameters of the Model," in: *Proceedings of 2015 International Conference on Mechanical Engineering and Electrical Systems,* Singapore, 16. – 18. 12. 2015, Francie: EDP Science, 2016, `http://doi.org/10.1051/matecconf/20164002005`

[A7] Kovalenko, I., Shynkarenko, A., a Garan, M., "Digital Magnifying Device Based on Prusa i3," in: *Proceedings of the International Conference on Applied*

Mechanics and Mechatronics Engineering, Bangkok, 25. – 26. 10. 2015, Lancaster: DEStech Publications, Inc., pp. 124-128, ISBN: 978-1-60595-021-1, 2015

Other publications:

[A8] Garan M., ”Identifikace dynamických systémů pomocí Bayesovských sítí,” in: *Competition ”Studentská vědecká a odborná činnost (SVOČ)”*, Liberec, 24. 05. 2018, Czech Republic: the Technical university of Liberec, 2018

[A9] Kovalenko, I., Taich, J., a Garan, M., ”Stavba 3D tiskárny na fotopolymery s kontrolou průběhu tisku,” in: *10th Annual International Conference 2017 Manufacturing Systems Today and Tomorrow,* Liberec, 9. - 10. 11. 2017, Czech Republic: Department of Manufacturing Systems and Automation, ISBN 978-80-7494-370-6, 2017

[A10] Garan, M. and Kovalenko, I., ”Generalized Algorithm for Estimation of Output from Linear Dynamic Systems with Derivatives of Input Signal and Non-Zero Initial Conditions,” in: *10th Annual International Conference 2017 Manufacturing Systems Today and Tomorrow,* Liberec, 9. - 10. 11. 2017, Czech Republic: Department of Manufacturing Systems and Automation, ISBN 978-80-7494-370-6, 2017

[A11] Garan, M.; Kovalenko, I.; Moučka, M.; Vagaská, A.: Electronic Emulator of Linear Dynamic Systems, Principia Cybernetica, 2015

[A12] Moučka, M.; Garan M.; Kovalenko, I.; Tóthová, M.: Návrh adaptivního regulátoru pro řízení systému na bázi pneumatických svalů, Principia Cybernetica, 2015

[A13] Garan, M., ”Estimation of Output from Linear Dynamical Systems Applying Runge-Kutta Method,” in: *Workshop Světlanka 2015*, Rokytnice nad Jizerou, 22. - 25. 9. 2015, Czech Republic: Technical University of Liberec, pp.205-209, ISBN: 978-80-7494-229-7, 2015

[A14] Garan, M., Moučka, M., ”Improvement of the Method for Measuring of High-Voltage Resistors’ Discharges,” in: *Workshop Světlanka 2014*, Rokytnice nad Jizerou, 16. - 19. 9. 2014, Czech Republic: Technical University of Liberec, pp.183-187, ISBN: 978-80-7494-100-9, 2014