KTH Electrical Engineering

Optimal input design for nonlinear dynamical systems: a graph-theory approach

Theory and Applications

PATRICIO E. VALENZUELA PACHECO

Licentiate Thesis
Stockholm, Sweden 2014

KTH Royal Institute of Technology
School of Electrical Engineering
Department of Automatic Control
SE-100 44 Stockholm
SWEDEN

TRITA-EE 2014:059
ISSN 1653-5146
ISBN 978-91-7595-339-7

Akademisk avhandling som med tillstånd av Kungliga Tekniska högskolan framläges till offentlig granskning för avläggande av teknologie licentiatesexamen i elektro och systemteknik fredagen den 21 november 2014 klockan 10.00 i Kollegiesalen, Kungliga Tekniska högskolan, Brinellvägen 8, Stockholm.

Tryck: Universitetsservice US AB

Abstract

Optimal input design concerns the design of an input sequence to maximize the information retrieved from an experiment. The design of the input sequence is performed by optimizing a cost function related to the intended model application. Several approaches to input design have been proposed, with results mainly on linear models. Under the linear assumption of the model structure, the input design problem can be solved in the frequency domain, where the corresponding spectrum is optimized subject to power constraints. However, the optimization of the input spectrum using frequency domain techniques cannot include time-domain amplitude constraints, which could arise due to practical or safety reasons.

In this thesis, a new input design method for nonlinear models is introduced. The method considers the optimization of an input sequence as a realization of the stationary Markov process with finite memory. Assuming a finite set of possible values for the input, the feasible set of stationary processes can be described using graph theory, where de Bruijn graphs can be employed to describe the process. By using de Bruijn graphs, we can express any element in the set of stationary processes as a convex combination of the measures associated with the extreme points of the set. Therefore, by a suitable choice of the cost function, the resulting optimization problem is convex even for nonlinear models. In addition, since the input is restricted to a finite set of values, the proposed input design method can naturally handle amplitude constraints.

The thesis considers a theoretical discussion of the proposed input design method for identification of nonlinear output error and nonlinear state space models. In addition, this thesis includes practical applications of the method to solve problems arising in wireless communications, where an estimate of the communication channel with quantized data is required, and application oriented closed-loop experiment design, where quality constraints on the identified parameters must be satisfied when performing the identification step.

Acknowledgements

The present work is the result of the support I received from many people. I would like to start by expressing my profound gratitude to my supervisors, Assistant Professor Cristian Rojas and Professor Håkan Hjalmarsson. You gave me the opportunity to continue my studies towards a Ph.D. degree, and I learned from you many important aspects to consider in research. During these years I enjoyed a lot the work we do in the system identification group, and I am pretty sure that this will be the case for the coming years, with new exciting challenges waiting for us!

The present form of this thesis is the contribution from many people who volunteer to read it and provide many valuable suggestions. Many thanks to Johan Dahlin, Christian Larsson, André Teixeira, Martin Jakobsson, and Håkan Terelius for the time you spent reading the first drafts of this thesis, and for the feedback you gave me to improve the document. I also want to thank the people who contributed to the articles that are part of this thesis. My profound gratitude also goes to Professor Thomas Schön, Professor Bo Wahlberg, Professor Brett Ninness, Dr. Juan Carlos Agüero, Dr. Boris Godoy, Johan Dahlin, and Afrooz Ebadat. Thank you very much!

I want to thank also the administrators Anneli Ström, Hanna Holmqvist, Kristina Gustafsson, Gerd Franzon, and Karin Karlsson for all the support I received from you to solve many different questions I came up with, and for the good times we spend during lunch (especially the episodes of *Lunch with Hanna*) and fika, I really appreciate that.

During my time in the Department I met many nice people, that contributes to make our working place even more enjoyable. I appreciate every conversation we have had, and all your support through these years. I would like to express my gratitude to all the people in the Department of Automatic Control, especially to Mohamed Rasheed Abdalmoaty, Mariette Annergren, Per Hägg, Christian Larsson, Niklas Everitt, Niclas Blomberg, Giulio Bottegal, Afrooz Ebadat, Riccardo Sven Risuleo, and Miguel Ramos for all the good times we have spent working together in the system identification group; Martin Jakobsson, Kuo-Yun Liang, Marco Molinari, Olle Trollberg, André Teixeira, Sadegh Talebi, Pedro Lima, and Demia Della Penda for making our office a nice working place, and all the time we shared talking about non-research topics; Damiano Varagnolo, Themistoklis Charalambous, Winston García, Farhad Farokhi, Euhanna Ghadimi, Bart Besselink,

P.G. di Marco, Martin Andreasson, Meng Guo, Stefan Magureanu, Arda Aytekin, Valerio Turri, and Burak Demirel for many interesting conversations; and Håkan Terelius for all the funny times we have had (including running, of course). Thanks a lot to all of you!

During my stay in Sweden I met very nice people who are willing to help when I need support. My profound gratitude goes to Mats and Awa Danielsson and their family; and Fernando de Hoyos and Lupita Nava for all the good times we have had in Stockholm. I am really indebted with you for all your love and support during our stay in Sweden, thank you a lot!

I also want to deeply thank to the academics in the Department of Electronic Engineering in my home university, Professors Mario Salgado, Eduardo Silva (rest in peace) and Ricardo Rojas, for guiding me in my first steps in the research field, and for encouraging me to continue my studies towards a Ph.D. degree. Thank you very much!

The step of doing a Ph.D. would not be possible without the support of many friends I met in Chile. I am really indebted with every one of you guys! My profound gratitude goes specially to Claudia Cortez, Marisol Vera, Alfred Rauch, Mauricio Moya, Cristian Carrasco, Felipe López, Sebastián "*el testigo*" Pulgar, Matías García and Gabriela Leal, Nelson López and Marcela Cubillos, Ramón Delgado, Rocío Guerra and their daughters, Diego Carrasco, Pedro Riffo and Valeria Araya, Iván Velásquez, Verónica Contreras and their children, Francisco Arredondo and Lucrecia Aedo, Stjpe Halat and Francisco Vargas. I really appreciate all the support I received from all of you over the years, thank you very much!

Finally, I would like to thank all the support and love I received from my family, my parents and my siblings. Your support and love is the essential energy I need to cope with all the challenges I face in my life. I am deeply indebted to my wife Daniela Medina for accepting the challenge of moving from Chile, and for all the love and support I receive from her every day, I love you! I am also really grateful to the love and support received from my parents Patricio Valenzuela and Brígida Pacheco, and my siblings Cristian (and his family), Alex and my little sister Claudia. My parents in law Germán Medina and Magda Miranda and my sisters in law Valentina and Rosario have always been present, giving us love and support through all these years. My gratitude goes also to Raúl Díaz, Teresa Ojeda and Mauricio Díaz (rest in peace), who are beyond a friendship: they became part of my family, and their love is always present. Thank you very much!

¡Muchas gracias de corazón a todos!
Patricio E. Valenzuela
Stockholm, Sweden. October 2014.

Contents

Glossary

$\delta(x)$	Dirac delta function at $x = 0$.
$(\cdot)^\top$	Transpose operator.
$\#\mathcal{X}$	Cardinality of the set $\mathcal{X}$.
$\det(\cdot)$	Determinant operator.
$\exp\{x\}$	Exponential function: e^x.
$\lambda_{\min}(\cdot)$	Minimum eigenvalue.
$\mathbf{1}_X$	Indicator function: $\mathbf{1}_X = 1$ if X is true; 0 otherwise.
$\mathrm{erf}(\mathcal{X})$	Error function: $\frac{2}{\sqrt{\pi}} \int_{\mathcal{X}} e^{-u^2} du$.
$\mathrm{supp}(p)$	Support of p.
$\mathrm{tr}\{\cdot\}$	Trace operator.
$\succeq$, $\preceq$	Matrix inequalities.
q	Time shift operator, $q\, u_t = u_{t+1}$.
$x_{1:N}$	$\{x_k\}_{k=1}^N$.
P	Cumulative distribution function (cdf).
p	Probability density function (pdf).
$\mathbf{E}\{\cdot\}$	Expectation operator.
$\mathrm{Cov}(x)$	$\mathbf{E}\left\{(x - \mathbf{E}\{x\})(x - \mathbf{E}\{x\})^\top\right\}$.
$\chi^2_\alpha(n)$	α-percentile of the χ^2-distribution with n degrees of freedom.
$\mathcal{N}(x, y)$	Normal distribution with mean x and variance $y \succeq 0$.
$\mathrm{As}\mathcal{N}(x, M)$	Asymptotic normal distribution with mean x and covariance $M \succeq 0$.
$\mathbf{P}\{\cdot\|\cdot\}$	Conditional probability measure.
$\mathbf{P}\{\cdot\}$	Probability measure.
$\mathcal{P}$	Set of cdfs associated with stationary vectors $\{u_t\}_{t=1}^{n_m}$.
$\mathcal{P}_\mathcal{C}$	Set of probability mass functions associated with stationary vectors $\{u_t\}_{t=1}^{n_m}$.
$\mathcal{V}_{\mathcal{P}_\mathcal{C}}$	Set of all the extreme points of $\mathcal{P}_\mathcal{C}$.
$\mathcal{E}$	Set of edges.
$\mathcal{V}$	Set of nodes.
$\mathcal{G}_\mathcal{V}$	Directed graph with nodes in $\mathcal{V}$.
$\mathcal{G}_{\mathcal{C}^n}$	n-dimensional de Bruijn graph derived from $\mathcal{C}^n$.

$\mathcal{A}_r$	Set of ancestors of r.
$\mathcal{D}_r$	Set of descendants of r.
$\mathcal{M}$	Model set.
Θ	Set of feasible parameters.
θ	Parameter employed for estimation.
θ_0	True parameter in Θ.
$\hat{\theta}_N$	Estimated parameter based on the data set $\mathbf{Z}^N$.
$\hat{y}_{t\|t-1}(\theta)$	Mean square optimal one-step ahead predictor given $\mathbf{Z}^{t-1}$.
$\mathcal{S}(\theta)$	Score function: $\dfrac{\partial}{\partial\theta}\log p_\theta(y_{1:N})$.
$\mathbf{Z}^N$	Data set composing of N samples.
$\mathbb{R}^+$	$\{x \in \mathbb{R} : x > 0\}$.
$\mathbb{R}^n$	Set of n-dimensional vectors with real entries.
$\mathbb{R}^{r\times s}$	Set of $r \times s$ matrices with real entries.
$\mathbb{Z}$	Set of integer numbers.

Acronyms

APF	Auxiliary particle filter.
AR	Autoregressive.
ARMA	Autoregressive moving average.
ARMAX	Autoregressive moving average with exogenous input.
ARX	Autoregressive with exogenous input.
cdf	Cumulative distribution function.
CRLB	Cramér-Rao lower bound.
EM	Expectation maximization.
FIM	Fisher information matrix.
FIR	Finite impulse response.
FL	Fixed-lag.
IS	Importance sampling.
MA	Moving average.
MIMO	Multiple input multiple output.
ML	Maximum likelihood.
MPC	Model predictive control.
NOE	Nonlinear output error.
NSSM	Nonlinear state space model.
OE	Output error.
pdf	Probability density function.
PEM	Prediction error method.
PF	Particle filter.
pmf	Probability mass function.

SIS	Sequential importance sampling.
SISO	Single input single output.
SMC	Sequential Monte Carlo.
SSM	State space model.

Chapter 1

Introduction

Modeling is an important stage in many applications. By modeling we mean the process of obtaining a mathematical description for a given natural phenomenon (also called *system*). Examples include prediction of prices in finance [85], channel estimation in communication systems [88], and design of controllers in industrial processes [80].

We can distinguish two main approaches to modeling. The first approach considers the modeling of a natural phenomenon based on physical laws (e.g., the use of the Kirchhoff's current and voltage laws to model an electrical circuit). The second approach is to model the natural phenomenon as a black box model (or gray box model, if some physical insight for the model is taken into account) and identify the model based on the input-output data from an experiment performed in the system (e.g., provide a voltage excitation to an electrical circuit, and measure the current to obtain a model for the equivalent impedance). In this thesis we are interested in the second approach, which is commonly referred to as *system identification*.

One key issue in system identification is the input sequence we provide to excite the system. Since system identification requires input-output data, it is relevant to design an input excitation that maximizes the information in the experiment in a certain sense (e.g., optimizing a cost function related to the intended model application). The process of designing an input sequence for system identification is commonly referred to as *input design*, which is the focus of this thesis.

A distinction is made between the terms input design and experiment design. In *experiment design*, the choices include the definition of input and output signals, the measurement instants, the manipulated signals, and how to manipulate them (which is the focus of input design). It also includes signal conditioning (e.g., the choice of presampling filters [59, Chapter 13]).

In this chapter, we provide some background on the theory of system identification and input design, which will be useful for the next chapters. In addition, this chapter also presents the main contributions of this work and the outline of the thesis.

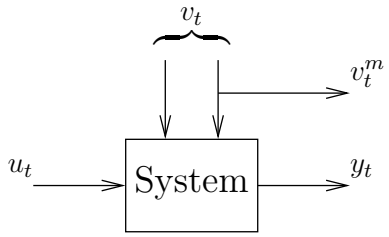

Figure 1.1: Block diagram representing the concept of a system.

1.1 System identification

System identification concerns the modeling of a system based on input-output data collected from it. Quoting [59, Section 1.1], we can define a *system* as "*an object in which variables of different kinds interact and produce observable signals*". In this context, the *inputs* are the external stimuli affecting the system (which are available to be manipulated by the user), and the *outputs* are the observable signals of interest. In addition, the system can also be affected by *disturbances*, which are stimuli that cannot be controlled (but which could possibly be measured). Figure 1.1 depicts a block diagram with the concept of a system. In this figure, y_t denotes the output, u_t the input, v_t^m the measured disturbances, and v_t the disturbances.

As can be seen from the given definition of a system, the theory of system identification can be applied in a wide range of fields. In the following subsections, we will provide a brief discussion of the main elements and results in the theory of system identification.

The basic entities

To build a model from data, we require three basic entities: a data set, a set of candidate models describing the relation between the input-output data (referred to as model structure), and an estimator to choose a model from the set of candidates (referred to as identification method).

Data set

In order to estimate a model, we require a set of input-output data from the system. This data set can also contain information regarding the measured disturbances. Following the notation in Figure 1.1, the data set is defined as $\mathbf{Z}^N := \{y_t, u_t, v_t^m\}_{t=1}^N$.

Model structure

Given the data set $\mathbf{Z}^N$, we want to use it to estimate a model for the system. The model is normally constrained to a set containing the possible mathematical descriptions, which is called the *model set*. The choice of the model set is based on prior information available about the system, on information obtained from the data set using nonparametric techniques, or a combination of both approaches. In the next examples we illustrate some common choices for the model set.

Example 1.1 *(Linear time invariant models) If the system is working locally around an equilibrium point, then it is reasonable to consider that it can be described by a model in the set*

$$\mathcal{M} = \{y_t = G(q;\,\theta)u_t + H(q;\,\theta)e_t | \theta \in \Theta\}\,, \tag{1.1}$$

where $\Theta \subseteq \mathbb{R}^{n_\theta}$ is a set of parameters. $G(q;\,\theta)$ and $H(q;\,\theta)$ are rational functions in the time shift operator q (i.e., $q\,u_t = u_{t+1}$), parameterized by θ. Here, $\{e_t\}$ is a white noise sequence with zero mean and finite variance. In this case, the disturbance v_t in Figure 1.1 can be described as

$$v_t = H(q;\,\theta_0)e_t\,, \tag{1.2}$$

for some $\theta_0 \in \Theta$. We notice that the system in Figure 1.1 is assumed to have additive output disturbances. Finally, since we do not have access to measure the disturbance (or part of it), the signal v_t^m in Figure 1.1 is empty.

Remark 1.1 *(Undermodeling) A common assumption on the model set $\mathcal{M}$ is that there exists at least one $\theta_0 \in \Theta$ such that the model evaluated at θ_0 describes the true system. If this condition is not fulfilled, then we say that the system is* undermodeled *by $\mathcal{M}$. In this thesis we assume that $\mathcal{M}$ contains the exact description of the system, i.e., there is no undermodeling.*

Depending on the assumptions on $G(q;\,\theta)$ and $H(q;\,\theta)$, the model set of linear systems in Example 1.1 can describe different structures, such as moving average (MA), autoregressive (AR), autoregressive with exogenous input (ARX), autoregressive moving average (ARMA), and autoregressive moving average with exogenous input ARMAX [59].

Example 1.2 *(Nonlinear state space models (NSSM)) Sometimes the linear assumption introduced in Example 1.1 is very restrictive. For example, the system could work in regions where the nonlinearities cannot be neglected.*

There are several alternatives to model nonlinear systems. One of the most general model sets is given in terms of a nonlinear state space description [77]. A nonlinear state space model is defined as

$$\mathcal{M} = \left\{ \begin{array}{l} x_{t+1} = f_\theta(x_t,\,u_t,\,e_t) \\ y_t = g_\theta(x_t,\,w_t) \end{array} \middle| \, \theta \in \Theta \right\}\,, \tag{1.3}$$

where $\Theta \subseteq \mathbb{R}^{n_\theta}$ is a set of parameters as in Example 1.1. f_θ and g_θ are nonlinear functions parameterized by θ. In this example, $\{e_t\}$ and $\{w_t\}$ are mutually independent white noise sequences, with zero mean and finite variance. In this case, v_t in Figure 1.1 is composed by the processes e_t and w_t, and the signal v_t^m is empty (as in Example 1.1).

We notice that the model set in Example 1.2 includes also linear time invariant models with static nonlinearities, known as *Wiener-Hammerstein models* [59]. Another interesting model set included in Example 1.2 is described in the following example.

Example 1.3 *(Nonlinear output error models (NOE)) A particular model set associated with the nonlinear state space model in Example 1.2 is*

$$\mathcal{M} = \left\{ \begin{array}{l} x_{t+1} = f_\theta(x_t,\, u_t) \\ y_t = h_\theta(x_t) + w_t \end{array} \;\middle|\; \theta \in \Theta \right\}. \tag{1.4}$$

Notice that (1.4) *can be obtained from* (1.3) *by setting $e_t = 0$ for all t, and $g_\theta(x_t,\, w_t) = h_\theta(x_t) + w_t$. The models in the set* (1.4) *will be referred to as* nonlinear output error *(NOE) models.*

We must emphasize that the model sets introduced in the previous examples are not the only ones existing in the literature. Indeed, it is possible to find model sets with time varying structure [59], neural networks [101], and models based on kernel estimators [67], among others. The results discussed in the following chapters will be mainly focused on the model sets introduced in Examples 1.1-1.3.

Identification method

A natural question is how to select a model from a given model set $\mathcal{M}$ and a given data set $\mathbf{Z}^N$. The goal is to choose the model from $\mathcal{M}$ that *best* explains the data $\mathbf{Z}^N$ in a given sense. The technique employed to choose a model from the structures in $\mathcal{M}$ is referred to as the *identification method*. Several identification methods have been proposed in the literature, including, e.g., least squares [28], instrumental variables [78], and subspace techniques [93], among others.

In this thesis, we work with two identification methods:

1. The maximum likelihood method, and

2. the prediction error method.

The methods are described in the next subsections.

The maximum likelihood method

The maximum likelihood (ML) method is one of the most attractive identification methods due to its statistical properties [24]. The ML method is based on the distribution of the data set $y_{1:N} := (y_1, \ldots, y_N)$, which is parameterized by $\theta \in \Theta$. The objective is to find the estimated parameter $\hat{\theta}_N$ that best explains the measurements $y_{1:N}$. If we define $p_\theta(y_{1:N})$ as the probability density function (pdf) associated with $y_{1:N}$, then the estimate $\hat{\theta}_N$ obtained by the ML method is given by

$$\hat{\theta}_N = \arg\max_{\theta\in\Theta} p_\theta(y_{1:N})\,. \tag{1.5}$$

The expression (1.5) has an intuitive interpretation: $\hat{\theta}_N \in \Theta$ is such that the observed event $y_{1:N}$ becomes "as likely as possible" [59].

For simplicity reasons, the logarithm of the probability density function (pdf) $p_\theta(y_{1:N})$ is usually maximized instead of $p_\theta(y_{1:N})$. This quantity is referred to as the *log-likelihood function*. Due to the monotonicity of the logarithm function, the solution

$$\hat{\theta}_N = \arg\max_{\theta\in\Theta} \log p_\theta(y_{1:N})\,, \tag{1.6}$$

is equal to the solution in (1.5). The expression (1.6) is usually prefered to (1.5) because products appearing in $p_\theta(y_{1:N})$ are converted into sums, and that the logarithm removes exponentials (when the density $p_\theta(y_{1:N})$ is in the exponential class [9]) since $\log\{e^a\} = a$. Another reason is that the use of logarithms results in algorithms that are numerically more well-behaved [76].

To illustrate the computation of the log-likelihood function, we consider the following example.

Example 1.4 *(ML of a nonlinear SSM) Consider the nonlinear SSM described in Example 1.2. Due to the stochastic properties of the sequences* $\{e_t\}$ *and* $\{w_t\}$*, the model set* (1.3) *can be rewritten as*

$$\mathcal{M} = \left\{ \begin{array}{l} x_{t+1} \sim p_\theta(x_{t+1}|x_t,\, u_t) \\ y_t \sim p_\theta(y_t|x_t) \\ x_1 \sim p_\theta(x_1) \end{array} \,\middle|\, \theta \in \Theta \right\}, \tag{1.7}$$

where "$\sim$*" denotes "*distributed according to*", and* $p_\theta(x_{t+1}|x_t,\, u_t)$*,* $p_\theta(y_t|x_t)$ *are the conditional probability density functions of* x_{t+1} *and* y_t*, conditioned on* x_t *and* u_t*, respectively.*

An important property associated with the stochastic models in (1.7) *is the* Markov property*, which means that the distribution of* x_{t+1} *and* y_t *given* $\{x_k,\, u_k\}_{k=-\infty}^{t}$ *equals their distribution given* $\{x_t,\, u_t\}$*.*

The likelihood function can be computed using the definition of conditional probability density functions as [69]

$$p_\theta(y_{1:N}) = p_\theta(y_1) \prod_{t=2}^{N} p_\theta(y_t|y_{1:t-1})\,. \tag{1.8}$$

Taking the logarithm of (1.8)*, we obtain*

$$\log p_\theta(y_{1:N}) = \log p_\theta(y_1) + \sum_{t=2}^{N} \log p_\theta(y_t|y_{1:t-1})\,. \tag{1.9}$$

From (1.9) *we see that the logarithm of the likelihood function transforms the product of pdfs into a sum.*

We use the Markov property associated with the model set (1.7) *to compute the pdfs in* (1.9) *as*

$$p_\theta(y_t|y_{1:t-1}) = \int_{\mathcal{X}_t} p_\theta(y_t|x_t)p_\theta(x_t|y_{1:t-1})\,dx_t\,, \tag{1.10}$$

$$p_\theta(x_t|y_{1:t}) = \frac{p_\theta(y_t|x_t)p_\theta(x_t|y_{1:t-1})}{p_\theta(y_t|y_{1:t-1})}\,, \tag{1.11}$$

$$p_\theta(x_{t+1}|y_{1:t}) = \int_{\mathcal{X}_t} p_\theta(x_{t+1}|x_t,\, u_t)p_\theta(x_t|y_{1:t})\,dx_t\,, \tag{1.12}$$

where $\mathcal{X}_t$ denotes the set of values for x_t. Equations (1.10)-(1.11) *are known as the* measurement update*, and equation* (1.12) *is known as the* time update*. Together, equations* (1.10)-(1.12) *can be employed to recursively compute the pdfs in the log-likelihood function* (1.9)*.*

Example 1.4 illustrates how the information available in the model set can be employed to compute the log-likelihood function. It is important to emphasize that, in general, equations (1.10)-(1.12) cannot be computed in closed form. An exception is when the system is linear and Gaussian, where we can recover the expressions for the Kalman filter [47] from (1.10)-(1.12). The reason is that the analytic solutions of the integrals (1.10) and (1.12) are only available for specific cases. When a closed-form expression is not available, the optimization of (1.9) over Θ is highly complex. An approach to solve this issue has been proposed in [77], where particle methods are employed to numerically compute the expectation-maximization algorithm [20, 66] to maximize (1.9).

The prediction error method

The prediction error method (PEM) is another approach to find the best model in the set $\mathcal{M}$ that explains the data $\mathbf{Z}^N$ [59]. In this method, the estimated parameter $\hat{\theta}_N$ is obtained as

$$\hat{\theta}_N = \arg\min_{\theta\in\Theta} V_N(\theta)\,, \tag{1.13}$$

where

$$V_N(\theta) := \sum_{t=1}^{N} \ell(\varepsilon_t(\theta))\,, \tag{1.14}$$

$$\varepsilon_t(\theta) := y_t - \hat{y}_{t|t-1}(\theta)\,. \tag{1.15}$$

Here $\hat{y}_{t|t-1}(\theta)$ denotes the mean square optimal one-step ahead predictor given $y_{1:t-1}$, which is computed as

$$\hat{y}_{t|t-1}(\theta) := \mathbf{E}\{y_t|y_{1:t-1}\} = \int_{\mathcal{Y}_t} y_t\, p_\theta(y_t|y_{1:t-1})\, dy_t\,, \tag{1.16}$$

and $\mathbf{E}\{\cdot\}$ is the expectation operator. In (1.16) $\mathcal{Y}_t$ denotes the set of values for y_t, and the function $\ell(\cdot)$ is an arbitrary and user chosen positive function (typically defined as a quadratic operator). We note that minimizing the prediction errors, $\varepsilon_t(\theta)$, makes sense since the models are normally employed for prediction, as in control system synthesis. Normally the systems are stochastic, which means that the output of the system at time t cannot be exactly determined by the data up to time $t-1$. Therefore, it is valuable to know at time $t-1$ what the output y_t is likely to be in order to compute the appropriate control action [79].

The prediction error method has a number of benefits [60]:

- It can be applied to a wide spectrum of model parameterizations since only an expression for (1.16) is required.
- It gives models with excellent asymptotic properties, thanks to its kinship with the ML method (cf. Example 1.5 below).
- It can handle systems that operate in closed-loop (the input is partly determined via output feedback, when the data are collected) without additional modifications to the method [27]. This property is also part of the ML method.

To illustrate the connection between PEM and the ML method, we introduce the following example:

Example 1.5 *(PEM and ML method) Consider a single output system that can be written as*

$$y_t = \hat{y}_{t|t-1}(\theta_0) + e_t\,, \tag{1.17}$$

where $\{e_t\}$ is white noise, Gaussian distributed with zero mean and variance $\sigma^2(\theta_0)$, and $\hat{y}_{t|t-1}(\theta_0)$ given by (1.16)*. Under the previous assumption, the log-likelihood function can be written as*

$$\log p_\theta(y_{1:N}) = -\frac{1}{2}\sum_{t=1}^{N} \varepsilon_t^2(\theta) - \frac{N}{2}\log\sigma^2(\theta) + c\,, \tag{1.18}$$

where $\varepsilon_t(\theta)$ is defined in (1.15)*, and c is a constant independent of θ. If we assume that $\varepsilon_t(\theta)$ and $\sigma^2(\theta)$ are independently parameterized in θ, then we can maximize*

over $\sigma^2(\theta)$ to obtain an expression in terms of $\varepsilon_t(\theta)$. This allows to rewrite (1.18) *as*

$$\log p_\theta(y_{1:N}) = -\frac{1}{N}\sum_{t=1}^{N} \varepsilon_t^2(\theta) + c\,. \tag{1.19}$$

If we compare (1.14) *with* (1.19)*, we see that PEM is retrieved from ML when $\{e_t\}$ is Gaussian distributed white noise, and $\ell(\varepsilon_t(\theta)) = N^{-1}\varepsilon_t^2(\theta)$.*

As in the previous subsection, equation (1.16) does not have a closed form expression in general. Except for linear and specific nonlinear models, expression (1.16) can only be computed numerically, e.g., using particle methods [77]. The next example shows the closed form expression for the optimal one-step ahead predictor in the linear case.

Example 1.6 *(Optimal one-step ahead predictor, linear case) Consider the model set $\mathcal{M}$ introduced in Example 1.1. We will compute its optimal one-step ahead predictor $\hat{y}_{t|t-1}(\theta)$, with $\theta \in \Theta$. To this end, we assume that $\lim_{q\to\infty} H(q;\,\theta) = I$. Under this assumption, we can rewrite any model in $\mathcal{M}$ as*

$$y_t = G(q;\,\theta)u_t + (H(q;\,\theta) - I)e_t + e_t\,. \tag{1.20}$$

We notice that $(H(q;\,\theta)-I)e_t$ in (1.20) *only contains information up to time $t-1$. Using* (1.1) *to compute e_t as a function of y_t and u_t, and inserting the result into* (1.20)*, we obtain*

$$y_t = H^{-1}(q;\,\theta)G(q;\,\theta)u_t + (I - H^{-1}(q;\,\theta))y_t + e_t\,. \tag{1.21}$$

The first two terms in the right-hand side of the equality in (1.21) *only depend on $\{y_k,\, u_k\}_{k=1}^{t-1}$ if G is strictly proper (otherwise it is not a problem if u_t is deterministic). In addition, since $\{e_t\}$ is a white noise sequence, we have that the best prediction of e_t given $\mathbf{Z}^{t-1}$ is $\mathbf{E}\{e_t\} = 0$. Therefore, the optimal one-step ahead predictor associated with the model set in Example 1.1 is*

$$\hat{y}_{t|t-1}(\theta) = H^{-1}(q;\,\theta)G(q;\,\theta)u_t + (I - H^{-1}(q;\,\theta))y_t\,. \tag{1.22}$$

The expression (1.22) *is a valid predictor if $H^{-1}(q;\,\theta)G(q;\,\theta)$ and $H^{-1}(q;\,\theta)$ are stable [59, 79].*

Remark 1.2 *In Example 1.6, e_t cannot be predicted from $\mathbf{Z}^{t-1}$ since e_t is white noise. Due to this property, e_t is called the* innovation *of the process [59].*

A useful predictor is introduced in the following example:

Example 1.7 *(Optimal one-step ahead predictor, nonlinear output error models) Consider the nonlinear output error model introduced in Example 1.3. Since $\{w_t\}$*

is a white noise sequence with zero mean and finite variance, the optimal one-step ahead predictor associated with this model is

$$x_{t+1} = f_\theta(x_t, u_t)\,, \tag{1.23a}$$
$$\hat{y}_{t|t-1}(\theta) = h_\theta(x_t)\,. \tag{1.23b}$$

As it can be seen from the previous examples, PEM can be applied to find the estimated parameters $\hat{\theta}_N$ for different models. The limitations of this method are that an expression for $\hat{y}_{t|t-1}(\theta)$ must be available, and that the optimal one-step ahead predictor $\hat{y}_{t|t-1}(\theta)$ must be stable. For general nonlinear models, it may happen that $\hat{y}_{t|t-1}(\theta)$ is difficult to compute. To circumvent this issue, it is possible to directly parameterize the model in terms of $\hat{y}_{t|t-1}(\theta)$ [59]. We notice that the direct parametrization of $\hat{y}_{t|t-1}(\theta)$ can be seen as a nonlinear output error model, introduced in Example 1.3.

Asymptotic analysis

An important question regarding identification methods is their *consistency*, i.e., if the identification method can retrieve the model in $\mathcal{M}$ describing the true dynamics of the system as $N \to \infty$. Furthermore, we want to know if $\mathbf{E}\{\hat{\theta}_N\} = \theta_0$, with $\theta_0 \in \Theta$ defined as the parameter describing the true system. An estimator $\hat{\theta}_N$ satisfying $\mathbf{E}\{\hat{\theta}_N\} = \theta_0$ is said to be *unbiased*.

On the other hand, we want to know the *accuracy* associated with the identification method, i.e., the size of the variation of the identified model around its expected value. In this subsection we briefly discuss the consistency and accuracy of the parameter estimates $\hat{\theta}_N$ as $N \to \infty$. To start the analysis, we require the following result [15, 59]:

Lemma 1.1 *(Cramér-Rao bound) Let $\hat{\theta}_N$ be an unbiased estimator of θ. Assume that $p_{\theta_0}(y_{1:N})$ (the pdf of $y_{1:N}$) is defined for all $\theta_0 \in \Theta$, and that for all the values of $y_{1:N}$ where $p_\theta(y_{1:N}) > 0$, we have that*

$$\frac{\partial}{\partial \theta} \log p_\theta(y_{1:N})\,, \tag{1.24}$$

exists, and that

$$\left| \frac{\partial}{\partial \theta_i} \log p_\theta(y_{1:N}) \right| \tag{1.25}$$

is bounded above by an integrable function over the set defined for $y_{1:N}$, for all $i \in \{1, \ldots, n_\theta\}$. In addition, suppose that $y_{1:N}$ may take values in a set whose boundaries do not depend on θ. Then

$$\mathbf{E}\left[\hat{\theta}_N - \theta_0\right]\left[\hat{\theta}_N - \theta_0\right]^\top \succeq \{\mathcal{I}_F^e\}^{-1}\,, \tag{1.26}$$

where

$$\begin{aligned}\mathcal{I}_F^e &:= \mathbf{E}\left\{ \frac{\partial}{\partial\theta}\log p_\theta(y_{1:N})\Big|_{\theta=\theta_0} \frac{\partial}{\partial\theta^\top}\log p_\theta(y_{1:N})\Big|_{\theta=\theta_0}\Big| u_{1:N}\right\} \\ &= -\mathbf{E}\left\{ \frac{\partial^2}{\partial\theta\partial\theta^\top}\log p_\theta(y_{1:N})\Big|_{\theta=\theta_0}\Big| u_{1:N}\right\}. \end{aligned} \tag{1.27}$$

The result introduced in Lemma 1.1 states that the covariance of any unbiased estimator $\hat{\theta}_N$ cannot be smaller than the inverse of $\mathcal{I}_F^e$, known as the *Fisher information matrix*. We notice that the computation of $\mathcal{I}_F^e$ requires the knowledge of $\theta_0 \in \Theta$, which implies that the exact value of $\mathcal{I}_F^e$ might not be available.

Remark 1.3 *The quantity*

$$\mathcal{S}(\theta) := \frac{\partial}{\partial\theta}\log p_\theta(y_{1:N}) \tag{1.28}$$

is usually known as the score function. *The score function will be employed in Chapter 4 to compute the Fisher information matrix for nonlinear state space models.*

To continue, we introduce the following definition:

Definition 1.1 *(Efficient and Asymptotically efficient estimators) An estimator $\hat{\theta}_N$ is said to be* efficient *if expression* (1.26) *holds with equality for all N. If the expression* (1.26) *holds with equality for $N \to \infty$, then $\hat{\theta}_N$ is said to be* asymptotically efficient.

The estimators obtained by the ML method and PEM (for a particular ℓ and Gaussian innovations) are asymptotically efficient. From this perspective, it is interesting to analyze how the estimator $\hat{\theta}_N$ behave as $N \to \infty$. The key result in this area was introduced in [15, 97] for the asymptotic distribution of maximum likelihood estimators obtained from independent observations:

Lemma 1.2 *(Consistency and asymptotic distribution of ML estimators) Suppose that the random variables $z_{1:N} := \{z_t\}_{t=1}^N$ are independent and identically distributed. Suppose also that the distribution of $z_{1:N}$ is given by p_{θ_0} for some value $\theta_0 \in \Theta$. Then, as $N \to \infty$, the random variable $\hat{\theta}_N$ tends to θ_0 with probability one, and the random variable $\sqrt{N}\left(\hat{\theta}_N - \theta_0\right)$ converges in distribution to*

$$\sqrt{N}\left(\hat{\theta}_N - \theta_0\right) \in \text{As}\mathcal{N}(0, \{\mathcal{I}_F^e\}^{-1}), \tag{1.29}$$

where $\mathcal{I}_F^e$ is given in Lemma 1.1.

The result introduced in Lemma 1.2 shows that, as $N \to \infty$, the distribution of the random variable $\sqrt{N}(\hat{\theta}_N - \theta_0)$ tends to be normal with zero mean and covariance matrix given by the Cramér-Rao bound. We notice that the result in Lemma 1.2 is also true when the ML method is applied to dynamical systems under some mild conditions. Moreover, under technical conditions, the result in Lemma 1.2 still holds for the estimator given by PEM when the white noise sequence $\{e_t\}$ is Gaussian and ℓ is a quadratic function [59].

Remark 1.4 *The asymptotic distribution given in Lemma 1.2 does not necessarily imply that*

$$\mathrm{Cov}(\sqrt{N}\hat{\theta}_N) := N\,\mathbf{E}\left\{(\hat{\theta}_N - \mathbf{E}\{\hat{\theta}_N\})(\hat{\theta}_N - \mathbf{E}\{\hat{\theta}_N\})^\top\right\} \to \{\mathcal{I}_F^e\}^{-1} \text{ as } N \to \infty\,. \tag{1.30}$$

The result (1.30) *requires more technical conditions on the stochastic processes acting on the system [59, Appendix 9B]. In this thesis we assume that those conditions are fulfilled, which allows us to write*

$$\mathrm{Cov}(\hat{\theta}_N) \approx \frac{1}{N}\{\mathcal{I}_F^e\}^{-1}\,. \tag{1.31}$$

As we can see in (1.31), the covariance matrix of an unbiased and asymptotically efficient estimator can be expressed in terms of the Cramér-Rao bound. One question that can arise at this point is if it is possible to *shape* the covariance matrix of $\hat{\theta}_N$ to improve the accuracy of the estimates. Indeed, since the Fisher information matrix (1.27) is conditioned on the input sequence $u_{1:N}$, it is possible to shape the covariance matrix of $\hat{\theta}_N$ by designing $u_{1:N}$, which is the main objective of input design, described in the next section.

1.2 Input design

Optimal input design concerns the design of an excitation that maximizes the information obtained in the data set $\mathbf{Z}^N$ [14, 23, 33]. The maximization is usually performed by optimizing a cost function related to the intended model application. Another standard choice for the cost function is a scalar number associated with the Fisher information matrix $\mathcal{I}_F^e$ [33, 45, 59]. We denote this cost function $h\colon \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}$. To obtain convex optimization problems for the input design techniques introduced in this thesis, h must satisfy the following definition [5]:

Definition 1.2 *(Matrix concave function) A function $f : \mathbb{R}^{r\times r} \to \mathbb{R}$ is called* matrix concave *if and only if, for every two matrices $X, Y \in \mathbb{R}^{r\times r}$ in the positive semidefinite cone, and for all $\lambda \in [0, 1]$,*

$$f(\lambda X + (1-\lambda)Y) \geq \lambda f(X) + (1-\lambda) f(Y)\,. \tag{1.32}$$

Table 1.1: Typical choices of h.

Optimality criterium	h
A-optimality	$-\mathrm{tr}\left\{(\cdot)^{-1}\right\}$
D-optimality	$\log\det(\cdot)$
E-optimality	$\lambda_{\min}(\cdot)$
L-optimality	$-\mathrm{tr}\left\{W(\cdot)^{-1}\right\}$

According to Definition 1.2, some suitable choices for h are $h = \log\det$ (D-criterion), $h = -\mathrm{tr}\left\{(\cdot)^{-1}\right\}$ (A-criterion), and $h = \lambda_{\min}$ (E-criterion), among others. Table 1.1 summarizes the definitions commonly used for the cost function h [49].

By designing an optimal input sequence for identification we mean that, for a given data length N, we optimize the accuracy for the parameter estimates in a prescribed sense. Since the cost function is associated with $\mathcal{I}_F^e$, then by Remark 1.4 we conclude that the maximization of $\mathcal{I}_F^e$ implies a reduction in the covariance matrix of $\hat{\theta}_N$. In practical applications, input design allows to reduce the time associated with the experiment to obtain a prescribed accuracy. To see this, we note that equation (1.31) implies that the covariance matrix of the parameter estimates decays as N^{-1}. Furthermore, if $\mathcal{I}_F^e$ is optimized, then from equation (1.31) we conclude that we can reduce the number of samples required to achieve the desired accuracy for $\hat{\theta}_N$.

To illustrate the importance of input design, we consider the following example:

Example 1.8 *(Fisher information matrix for an FIR model) Consider the finite impulse response (FIR) model*

$$y_t = \theta_1 u_t + \theta_2 u_{t-1} + e_t\,, \tag{1.33}$$

where $\theta = \begin{bmatrix}\theta_1 & \theta_2\end{bmatrix}^\top \in \mathbb{R}^2$*, and* $\{e_t\}$ *is Gaussian white noise with zero mean and variance* λ_e*. We are interested in identifying* $\theta \in \mathbb{R}^2$ *by performing an experiment with* $N = 2$ *samples and* $u_t \in \{1, 0\}$*. The Fisher information matrix* $\mathcal{I}_F^e$ *for the model* (1.33) *is (assuming* $u_0 \neq 0$*)*

$$\mathcal{I}_F^e = \frac{1}{\lambda_e}\begin{bmatrix} u_1^2 + u_2^2 & u_2\,u_1 + u_1\,u_0 \\ u_2\,u_1 + u_1\,u_0 & u_1^2 + u_0^2 \end{bmatrix}. \tag{1.34}$$

From (1.34) *we see that* $\mathcal{I}_F^e$ *depends on the input samples* $\{u_0, u_1, u_2\}$*. Therefore, the choice of the input for the experiment is crucial to identify* θ*. For example, if* $u_1 = u_2 = 1$ *and* $u_0 = 1$*, the matrix* $\mathcal{I}_F^e$ *becomes*

$$\mathcal{I}_F^e = \frac{1}{\lambda_e}\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}, \tag{1.35}$$

which is singular. This implies that the parameter θ cannot be fully identified. Indeed, if we let the number of samples $N \to \infty$, a constant input signal can only be employed to identify the DC gain of the model (1.33)*, which is the sum of the parameters. However, if in the previous case we choose $u_1 = 0$, then $\mathcal{I}_F^e$ results in*

$$\mathcal{I}_F^e = \frac{1}{\lambda_e} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tag{1.36}$$

which is a nonsingular matrix. In conclusion, the design of $\{u_1, u_2\}$ to identify the model (1.33) *is an important step to obtain the desired results.*

The optimal input design problem has been widely analyzed in the literature, with numerous results available. In the next subsection we provide a literature review with the main results in input design.

Literature review on input design

Most results in input design for dynamical systems have been developed for linear models [59]. The assumption of a linear model structure allows to use convex optimization tools to solve the input design problem [32, 45, 57, 59, 74]. Several methods to design inputs for identification of linear systems have been reported in the literature. In [74] the problem of input design for identification of linear systems is addressed. The input sequence in [74] is designed by optimizing the experiment for the worst case scenario defined by the model parameters, which are assumed to lie in a given compact set. The optimal input design in [74] also considers energy (or power) constraints for the input sequence. Moreover, [74] shows a convex optimization algorithm that can be employed to solve a discretized approximation to the design problem. Another possibility to solve the optimal input design problem is to employ linear matrix inequalities (LMI) to characterize autocovariance functions associated with a feasible input spectrum [45, 57, 75, 96]. In [45] the input design problem is solved in the frequency domain, where the input spectrum is designed. To this end, [45] parameterizes the input spectrum using rational basis functions. This allows to obtain a convex problem in the decision variables, where quality constraints on the identified model, and power constraints on the input signal can be included. A D-optimal multisine excitation is designed in [75], where the signal is employed to model physiological or electrochemical phenomena from spectroscopy measurements. In [96] the optimal input design is presented for finite impulse response (FIR) models, minimizing the uncertainty of the identified model while the variance of the input is kept as small as possible. A Markov chain approach is presented in [6, 7] to design inputs with amplitude constraints. The input sequence is assumed to be the output of a Markov chain, where the transition probability matrix is designed to maximize the information retrieved from the experiment. However, the resulting problem is non-convex and numerical optimization tools must be employed. We find the same problem for

time domain gradient-based schemes [32, 83], where only the convergence to local optima can be guaranteed.

In recent years, the interest on input design has been extended from linear to nonlinear systems. The main issue here is that most of the tools used for input design for linear systems based on frequency domain techniques are no longer valid for the nonlinear case. One approach to input design for nonlinear systems is introduced in [42], where a linear systems perspective is considered. Based on a particular nonlinear system, [42] raises the issue of obtaining a parametrization for the input sequence that results in a tractable optimization. The main message in [42] is that it is possible to reuse some of the parameterizations of input sequences for linear models to the nonlinear case. Thus, it is possible to parameterize the input sequence in terms of a probability density function describing a stationary stochastic process, but it is not straightforward to parameterize the input sequence in terms of its autocovariance function. In addition, [42] shows that it is possible to use the sum-of-squares method to relax the input design problem for nonlinear models. Extensions to a class of FIR type systems is developed in [54], where a characterization of probability density functions is employed. The assumption in [54] is that the input sequence is a realization of a stationary process with finite memory. Taking into account the structure of nonlinear FIR models, in [54] it is shown that the requirement of stationary process for the input can be combined with the model structure to obtain a convex optimization problem. Input design for structured nonlinear identification is introduced in [94, 95], where the system is assumed to be an interconnection of linear systems and static nonlinearities. The objective in [94, 95] is to minimize the variance of the experiment, while achieving the desired accuracy in the parameter estimates. It is shown in [94, 95] that the optimization problem can be expressed in terms of the probability mass function characterizing the input sequence. Moreover, [94, 95] shows that the resulting optimization problem is convex in the decision variables. Once the optimal probability mass function is obtained, [94, 95] generate an input realization using elements from graph theory. An input design method for nonlinear state space models is presented in [34], where a particle filter is used to approximate the cost function associated with the input design problem, which is optimized over a particular class of input vectors using stochastic approximation. In [34] it is assumed that the input sequence is an autoregressive process filtering a white noise process with prescribed variance, and the parameters of this process are optimized numerically. The methods previously mentioned [34, 42, 54, 94, 95] are in general highly complex (usually non-convex problems, e.g., [34]) and are restricted to particular model structures (e.g., [42, 54, 94, 95]) or particular classes of input signals (e.g., white noise filtered through an ARX filter [34]). Moreover, except for the results in [6, 7, 54], the methods introduced cannot handle input design with amplitude constraints. Amplitude constraints can arise due to safety reasons or physical limitations in the system. Therefore, input design with amplitude constraints even for linear systems also requires further study.

1.3 Thesis outline and main contributions

In this thesis we present a novel approach to input design for nonlinear systems. The approach considers the design of input sequences for models with additive white noise at the output, and nonlinear state space models, extending the class of nonlinear systems considered in [54]. The input is constrained to be a stationary process with a finite set of possible values, and the associated probability mass function (pmf) to have finite memory, i.e., a Markov chain of fixed order. Therefore, the problem is to find the pmf that maximizes the information obtained from the experiment, quantified as a scalar function of the information matrix. By using notions of graph theory, we can express the feasible set of pmfs as a convex combination of the pmfs of the so-called prime cycles describing the vertices of the set. Since the prime cycles can be explicitly computed by known algorithms [46, 100], the optimization problem becomes easy to pose. Furthermore, for standard choices of the cost function, the problem is convex even for nonlinear systems, which reduces the computational complexity compared with the Markov chain approach in [6, 7]. Finally, since the input is restricted to a finite set of possible values, the method naturally incorporates amplitude constraints.

The proposed input design method has proven to be an interesting solution to practical problems arising in the identification and estimation literature. As an illustration of the practical relevance of the proposed method, this thesis includes two practical applications. First, we employ the input design technique for channel estimation with quantized output data. In this problem the quantized output data introduces nonlinear behavior, which restricts the techniques that can be employed to design an input sequence. Second, the proposed input design method is used in closed-loop systems, where an external excitation is designed by minimizing the experimental cost, subject to probabilistic constrains on the input and output of the plant, and simultaneously achieving a prescribed accuracy for the identified model. In this case, the cost function differs from the one employed in the previous discussion, since we aim to minimize the experimental cost instead of maximizing the information in the experiment. However, the accuracy for the identified model appears as a constraint, which guarantees the desired results in the solutions of the optimization problem.

This thesis is organized in seven chapters, and 3 appendices. The chapters are structured into two parts. In the first part, we discuss the theory employed by the input design method, and the proposed solution to the problem of input design for nonlinear model structures. The second part describes the applications where the method has been employed to improve the accuracy of the identified parameters.

The contents in each chapter are as follows:

Chapter 2: This chapter introduces notions from graph theory and their relations to stationary processes. We show how the class of so-called de Bruijn graphs can be employed to describe the set of stationary processes with finite memory and finite alphabet. The contents in this chapter are partially based on

P.E. Valenzuela, C.R. Rojas, and H. Hjalmarsson. A graph theoretical approach to input design for identification of nonlinear dynamical models. *Accepted for publication*, Automatica, 2014.

P.E. Valenzuela, C.R. Rojas, and H. Hjalmarsson. Optimal input design for dynamic systems: a graph theory approach. *In proceedings of the* 52^{nd} *Conference on Decision and Control (CDC)*, Florence, Italy, 2013.

Chapter 3: In this chapter a method for input design for nonlinear output error models is presented. The method considers the design of an input sequence as a realization of a stationary process with finite memory and finite alphabet, which allows us to use the theory introduced in Chapter 2 to obtain a tractable problem. The results in this chapter are based on

P.E. Valenzuela, C.R. Rojas, and H. Hjalmarsson. A graph theoretical approach to input design for identification of nonlinear dynamical models. *Accepted for publication*, Automatica, 2014.

P.E. Valenzuela, C.R. Rojas, and H. Hjalmarsson. Optimal input design for dynamic systems: a graph theory approach. *In proceedings of the* 52^{nd} *Conference on Decision and Control (CDC)*, Florence, Italy, 2013.

Chapter 4: In this chapter an extension of the input design method to nonlinear state space models is presented. The method considers the approach introduced in Chapter 2. In this chapter the information matrix is computed as the sample covariance of the score function, which is approximated using particle methods. The results in this chapter are based on

P.E. Valenzuela, J. Dahlin, C.R. Rojas, and T.B. Schön. A graph/particle-based method for experiment design in nonlinear systems. *In proceedings of the* 19^{th} *IFAC World Congress*, Cape Town, South Africa, 2014.

Chapter 5: In this chapter we discuss an application of the input design method to channel estimation with quantized output measurements. Using an available expression for the information matrix of systems with quantized output, we use the input design method based on graph theory to design an input sequence to identify the model. The results in this chapter are based on

B.I. Godoy, P.E. Valenzuela, C.R. Rojas, J.C. Agüero, and B. Ninness. A novel input design approach for systems with quantized output data. *In proceedings of the* 13^{th} *European Control Conference (ECC)*, Strasbourg, France, 2014.

Chapter 6: In this chapter an application of the input design method to closed-loop identification is presented. The objective is to design an input sequence as a realization of a stationary process to identify a system operating in closed loop. The designed input sequence satisfies requirements on the information retrieved from the experiment, and probabilistic bounds on the input and

output of the system. Using the proposed input design method, the resulting problem is convex in the decision variables. The results in this chapter are based on

> A. Ebadat, P.E. Valenzuela, C.R. Rojas, H. Hjalmarsson, and B. Wahlberg. Applications oriented input design for closed-loop system identification: a graph-theory approach. *Accepted for publication in the* 53^{rd} *Conference on Decision and Control (CDC)*, Los Angeles, United States, 2014.

Chapter 7: This chapter presents the conclusion of the thesis and future work on the subject.

As a complement to the discussion in the chapters, 3 appendices are included:

Appendix A: We provide algorithms to compute the elementary cycles of a given graph.

Appendix B: We provide the proof of the convergence for the approximation of the Fisher information matrix employed in Chapter 3.

Appendix C: We review the expectation-maximization algorithm, and derive some identities that are useful in this thesis.

Part I

Theory

Chapter 2

Graph theory and stationary processes

The results presented in this thesis rely on the connection between stochastic processes and graph theory. Therefore, the relation between these two concepts must be clarified before presenting the results in the coming chapters.

In this chapter, we introduce the elements from graph theory required to understand the discussions in the next chapters. We also define de Bruijn graphs, and how they can be associated with stationary processes of finite memory. For this purpose, we describe the set of stationary processes of a given memory as a convex combination of the measures associated with the *prime cycles* of a de Bruijn graph [100].

Given an element in the set of stationary processes with finite memory, we want to sample a realization with the prescribed distribution. In this chapter, we also provide a novel method to obtain samples from a given probability measure that describes the stationary distribution of a Markov chain.

2.1 Graph theory: basic concepts

In this section we provide a number of definitions for the graph theory concepts employed in the next chapters. Our notation follows that of [46, pp. 77].

Definition 2.1 *(Directed graph) A* directed graph $\mathcal{G}_{\mathcal{V}} = (\mathcal{V}, \mathcal{E})$ *is a pair consisting of a nonempty and finite set of vertices (called* nodes*)* $\mathcal{V}$ *and a set* $\mathcal{E}$ *of ordered pairs* (v_i, v_j) *of vertices* $v_i, v_j \in \mathcal{V}$ *called* edges.

Remark 2.1 *We note that Definition 2.1 does not impose restrictions over the set of vertices* $\mathcal{V}$*. This allows to define* $\mathcal{V}$ *according to the requirement of the user.*

Definition 2.2 *(Path) A* path *in* $\mathcal{G}_{\mathcal{V}}$ *is a sequence of vertices*

$$p_{vu} = (v_1 = v, v_2, \ldots, v_k = u)\,,$$

such that $(v_i, v_{i+1}) \in \mathcal{E}$ *for all* $i \in \{1, \ldots, k-1\}$.

Definition 2.3 *(Cycle and elementary cycle) A* cycle *is a path in which the first and last vertices are identical. A cycle is* elementary *if no vertex except the first and last appears twice.*

Definition 2.4 *(Cyclic permutation) Consider two cycles* p_{uu} *and* p_{vv} *in* $\mathcal{G}_{\mathcal{V}}$. *We say that* p_{vv} *is a* cyclic permutation *of* p_{uu} *if and only if there exists paths* p_{uv}, *and* p_{vu} *such that* $p_{vv} = (p_{vu}, p_{uv})$, *where the first element in* p_{uv} *is removed, and* $p_{uu} = (p_{uv}, p_{vu})$, *where the first element in* p_{vu} *is removed.*

Definition 2.5 *(Distinct elementary cycles) Two elementary cycles are* distinct *if one is not a cyclic permutation of the other.*

De Bruijn graphs

The results in this thesis are based on the class of directed graphs, called *de Bruijn* graphs [17]. Their definition is given below.

Definition 2.6 *(de Bruijn graph) An* n*-dimensional* de Bruijn graph *of* m *symbols* $\mathcal{C} = \{s_1, \ldots, s_m\}$ *is a directed graph whose set of vertices* $\mathcal{V}$ *is given by*

$$\begin{aligned}\mathcal{V} = \mathcal{C}^n = \{&v_1 = (s_1, \ldots, s_1, s_1),\, v_2 = (s_1, \ldots, s_1, s_2), \ldots,\\ &v_m = (s_1, \ldots, s_1, s_m),\, v_{m+1} = (s_1, \ldots, s_2, s_1), \ldots,\\ &v_{m^n} = (s_m, \ldots, s_m, s_m)\}\,,\end{aligned} \tag{2.1}$$

and whose set of directed edges $\mathcal{E}$ *is*

$$\mathcal{E} = \{((v_1, r_1, \ldots, r_{n-1}), (r_1, r_2, \ldots, r_n)) : v_1, r_1, \ldots, r_n \in \mathcal{C}\}\,. \tag{2.2}$$

Remark 2.2 *According to Definition 2.6, an* n*-dimensional de Bruijn graph of* m *symbols is a directed graph representing overlaps between sequences of symbols. It has* m^n *vertices, consisting of all possible sequences of length* n *derived from the given symbols of length* n. *The same symbol can appear multiple times in a sequence. Moreover, if one of the vertices can be expressed as another vertex by shifting all its symbols one place to the left and adding a new symbol at the end, then the latter has a directed edge to the former vertex.*

To illustrate Definition 2.6 we consider the following example:

Example 2.1 *Consider a sequence* $\{u_t\}$ *with alphabet* $\mathcal{C} = \{0, 1\}$. *We are interested in deriving the de Bruijn graph associated with the possible transitions of* (u_{t-1}, u_t) *among its states in* $\mathcal{C}^2$. *In other words, we want to derive the possible transitions when we move from* (u_{k-1}, u_k) *to* (u_k, u_{k+1}). *Table 2.1 contains all the possible transitions between the elements in* $\mathcal{C}^2$. *We notice that the number of possible values of* (u_k, u_{k+1}) *for a given* (u_{k-1}, u_k) *is two, which is the number of*

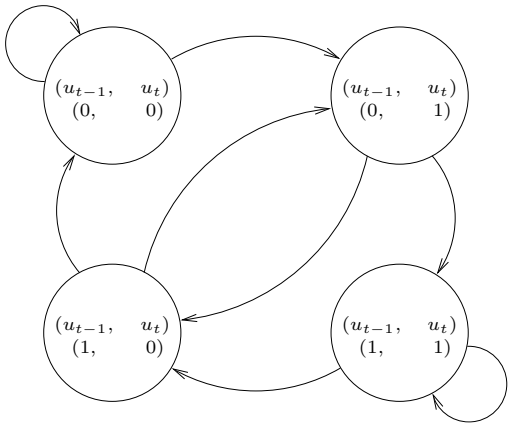

Figure 2.1: The de Bruijn graph derived from $\mathcal{C}^2$, with $\mathcal{C} = \{0, 1\}$.

(u_{t-1}, u_t)	(u_t, u_{t+1})
(0, 0)	{(0, 0), (0, 1)}
(0, 1)	{(1, 0), (1, 1)}
(1, 0)	{(0, 0), (0, 1)}
(1, 1)	{(1, 0), (1, 1)}

Table 2.1: Transitions from (u_{t-1}, u_t) to (u_t, u_{t+1}), Example 2.1.

elements in $\mathcal{C}$. This is due to the fact that the new element u_{k+1} belongs to $\mathcal{C}$. The resulting de Bruijn graph is depicted in Figure 2.1. We notice that the nodes are given by the elements of $\mathcal{C}^2$, and the edges correspond to the transitions presented in Table 2.1.

Remark 2.3 *We use $\mathcal{G}_{\mathcal{C}^n}$ to denote the n-dimensional de Bruijn graph derived from $\mathcal{C}^n$.*

2.2 De Bruijn graphs and stationary processes

In this section, we establish the connection between de Bruijn graphs and stationary processes. Before proceeding, we introduce the following definitions:

Definition 2.7 *(Cumulative distribution function) A function $P\colon \mathcal{C}^{n_m} \to \mathbb{R}$ is a* cumulative distribution function *(cdf) if and only if $P(x_1, \ldots, x_{n_m}) = \mathbf{P}\{X_1 \leq x_1, \ldots, X_{n_m} \leq x_{n_m}\}$, where $\{X_i\}_{i=1}^{n_m}$ are random variables, and $\mathbf{P}$ is a probability measure.*

Definition 2.8 *(Stationary process) Consider the stochastic process* $\{u_t\}$*. We say that the process* $\{u_t\}$ *is* stationary *if and only if the cumulative distribution functions* P *associated with* $\{u_t\}$ *satisfy* $P(u_{i:i+k}) = P(u_{j:j+k})$ *for all integers* i*,* j*,* k*.*

As we can see from Definition 2.8, a stationary process requires that all its marginal cumulative distribution functions (cdfs) are time invariant. The same definition can be restricted to the case of vectors:

Definition 2.9 *(Stationary vectors) Consider the vector* $u_{1:n_m}$*. We say that the vector* $u_{1:n_m}$ *is* stationary *if and only if the cdf associated with* $u_{1:n_m}$ *satisfy*

$$P(u_{i:i+k}) = P(u_{j:j+k})$$

for all positive integers i*,* j*,* k *satisfying* $1 \leq i \leq j \leq n_m - k$*.*

Stationary vectors

The concepts introduced in Section 2.1 can be employed to describe the set of stationary processes. To this end, we consider the sequence $\{u_t\}$, where $u_t \in \mathcal{C}$. Before continuing, we introduce the following definition:

Definition 2.10 *(Probability mass function) A* probability mass function *(pmf)* p *is a probability measure whose support is a set with finite cardinality.*

In this thesis we are interested in the case where $\{u_t\}$ is a stationary process. The description of $\{u_t\}$ is given in terms of the probability mass function (pmf), denoted by p. However, the description of $\{u_t\}$ as stationary process is intractable since we need to define an infinite number of cumulative distribution functions (cf. Definition 2.8). To solve this issue, we use the following definitions:

Definition 2.11 *(Extension of the cdf) Consider the cdf* $P(u_{1:n_m})$*. The* extension *of the cdf* $P(u_{1:n_m})$ *is defined as a Markov process* $\{u_t\}$ *of order* n_m*, whose stationary distribution is given by* $P(u_{1:n_m})$*.*

Definition 2.12 *(*n_m*-dimensional projection [100, Theorem 1]) Consider the cdf* $P(u_{1:n_m})$*. An* n_m-dimensional projection $P(u_{1:n_m})$ *of* $P(u_{1:n_{\text{seq}}})$ *(*$n_{\text{seq}} \geq n_m$*) is a cdf* $P(u_{1:n_m})$ *associated with the stationary vector* $u_{1:n_m}$*, which can be extended to the stationary vector* $u_{1:n_{\text{seq}}}$ *to define* $P(u_{1:n_{\text{seq}}})$*.*

Based on Definitions 2.11-2.12, we assume the following:

Assumption 2.1 *The stationary process* $\{u_t\}$ *is an extension of the stationary cumulative distribution function* $P(u_{1:n_m})$*.*

Assumption 2.1 restricts the set of stationary processes $\{u_t\}$ to those that can be obtained as an extension of pdfs with a finite number of elements. This means that $P(u_{1:n_{\text{seq}}})$ can be completely described by its n_m-dimensional projection $P(u_{1:n_m})$ for any $n_{\text{seq}} \geq n_m$ [100].

Given the restriction to the family of stationary processes we are interested in, we need to find a description for the set of pdfs associated with stationary processes of finite memory. For this end, we introduce the following result in terms of the cdf (whose proof follows from [100]):

Lemma 2.1 *(Shift invariant property) Consider a stationary vector $u_{1:n_m}$. A cdf $P\colon \mathbb{R}^{n_m} \to \mathbb{R}$ is a valid cdf for $u_{1:n_m}$ if and only if, for all $\mathbf{z} \in \mathbb{R}^{n_m-1}$,*

$$\int_{v\in\mathbb{R}} dP\left(\left[v,\, \mathbf{z}\right]\right) = \int_{v\in\mathbb{R}} dP\left(\left[\mathbf{z},\, v\right]\right) . \tag{2.3}$$

Proof *We know that P is a valid cdf for the stationary vector $u_{1:n_m}$ if and only if $u_{i:i+k} \overset{d}{\sim} u_{j:j+k}$, for all positive integers i, j, k satisfying $1 \leq i \leq j \leq n_m - k$. Here, $\overset{d}{\sim}$ denotes equal in distribution.*

First we assume that P is a valid cdf for the stationary vector $u_{1:n_m}$. This implies that $u_{1:n_m-1} \overset{d}{\sim} u_{2:n_m}$, which is equivalent to the condition written in (2.3). To prove the converse, we assume that (2.3) is true. Therefore, $u_{1:n_m-1} \overset{d}{\sim} u_{2:n_m}$ is satisfied. Then for any $k < n_m$, and by successive shifts, the marginal cdf $P(u_{1:k})$ obtained from $P(u_{1:n_m})$ satisfies

$$u_{1:k} \overset{d}{\sim} u_{2:k+1} \overset{d}{\sim} u_{3:k+2} \ldots \overset{d}{\sim} u_{n_m-k+1:n_m} , \tag{2.4}$$

which implies that $u_{1:n_m}$ has a stationary distribution. ■

Lemma 2.1 states that a cdf P is associated with a stationary vector $u_{1:n_m}$ if and only if the cdf obtained by marginalizing over u_{n_m} is equivalent to the one obtained by marginalizing over u_1. If a cdf P satisfies this property, then we say that P is *shift invariant*.

The result introduced in Lemma 2.1 allows us to characterize the set of cdfs associated with stationary vectors $u_{1:n_m}$ as

$$\begin{aligned}\mathcal{P} := \Bigg\{ P\colon \mathbb{R}^{n_m} \to \mathbb{R} \,|\, & P(\mathbf{x}) \geq 0,\, \forall \mathbf{x} \in \mathbb{R}^{n_m}; \\ & P \text{ is monotone nondecreasing ;} \\ & \lim_{\substack{x_i\to\infty \\ i\in\{1,\ldots,n_m\}}} P(x_1, \ldots, x_{n_m}) = 1, \\ & \int_{v\in\mathbb{R}} dP\left(\left[v,\, \mathbf{z}\right]\right) = \int_{v\in\mathbb{R}} dP\left(\left[\mathbf{z},\, v\right]\right) ,\forall \mathbf{z} \in \mathbb{R}^{n_m-1} \Bigg\} . \end{aligned} \tag{2.5}$$

We notice in (2.5) that the first three properties define a valid cdf. Introducing shift invariance as the fourth property in the set $\mathcal{P}$, we obtain the projection of the cdfs of stationary sequences into the set of stationary vectors of length n_m [100].

To exploit the relation between de Bruijn graphs and stationary vectors, we assume the following:

Assumption 2.2 *The alphabet $\mathcal{C}$ is a finite set with cardinality $n_{\mathcal{C}}$.*

In this thesis we restrict the analysis to the case when Assumption 2.2 is satisfied. Therefore, the set defined in Equation 2.5 will be restricted to the case of pmfs. Thus, the set we employ here is given by

$$\mathcal{P}_{\mathcal{C}} := \{p\colon \mathcal{C}^{n_m} \to \mathbb{R} |\, p(\mathbf{x}) \geq 0,\, \forall \mathbf{x} \in \mathcal{C}^{n_m};\; \sum_{\mathbf{x}\in\mathcal{C}^{n_m}} p(\mathbf{x}) = 1;\; \sum_{v\in\mathcal{C}} p\left([v,\, \mathbf{z}]\right) = \sum_{v\in\mathcal{C}} p\left([\mathbf{z},\, v]\right)\, , \forall \mathbf{z} \in \mathcal{C}^{n_m-1} \right\}. \quad (2.6)$$

Remark 2.4 *The cdfs associated with the elements in $\mathcal{P}_{\mathcal{C}}$ define a proper subset of $\mathcal{P}$.*

We need to parameterize the elements of $\mathcal{P}_{\mathcal{C}}$ in a tractable manner. This issue is relevant since in the next chapters we will optimize cost functions defined in terms of a stationary vector $\{u_t\}_{t=1}^{n_{\text{seq}}}$, which are described as an extension of the pmfs $p(u_{1:n_m}) \in \mathcal{P}_{\mathcal{C}}$, where $n_m < n_{\text{seq}}$. A natural approach is to parameterize $p(u_{1:n_{\text{seq}}})$ in terms of a finite number of elements of $\mathcal{P}_{\mathcal{C}}$, i.e., projections of stationary pmfs, since these elements can be extended to a full stationary pmf $p(u_{1:n_{\text{seq}}})$. Such a parameterization is described in the next subsection.

Parameterization of $\mathcal{P}_{\mathcal{C}}$

To parameterize the elements of $\mathcal{P}_{\mathcal{C}}$, we first notice that $\mathcal{P}_{\mathcal{C}}$ is a convex set, and, in particular, a polyhedron [72, pp. 170], since it is described by a finite number of linear equalities and inequalities. Hence, any element of $\mathcal{P}_{\mathcal{C}}$ can be described as a convex combination of its extreme points [72, Corollaries 18.3.1 and 19.1.1]. In other words, if we define $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}} := \{w_i, i = 1, ..., n_{\mathcal{V}}\}$ as the set of all the extreme points of $\mathcal{P}_{\mathcal{C}}$, then for all $f \in \mathcal{P}_{\mathcal{C}}$ we have

$$f = \sum_{i=1}^{n_{\mathcal{V}}} \alpha_i\, w_i\,, \quad (2.7)$$

where $\alpha_i \geq 0$, $i \in \{1, \ldots, n_{\mathcal{V}}\}$, and

$$\sum_{i=1}^{n_{\mathcal{V}}} \alpha_i = 1\,. \tag{2.8}$$

The set $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$ can be characterized in a graph-theoretical manner. Since we have restricted u_t to belong to a finite alphabet $\mathcal{C}$ with $n_{\mathcal{C}}$ elements (see Assumption 2.2), we notice that the set of possible values for $(u_{t-n_m+1}, \ldots, u_t)$, $\mathcal{C}^{n_m}$, is composed of $n_{\mathcal{C}}^{n_m}$ elements, which can be viewed as nodes in a graph. In addition, the transitions between the elements in $\mathcal{C}^{n_m}$, as described by a stationary process of memory n_m, are given by the possible values of u_{t+1} when we move from $(u_{t-n_m+1}, \ldots, u_t)$ to $(u_{t-n_m+2}, \ldots, u_{t+1})$, for all integers $t \geq 0$. The edges between the elements in $\mathcal{C}^{n_m}$ denote the possible transitions between the states, represented by the nodes of the graph. The resulting graph corresponds to a de Bruijn graph. To illustrate the transition among the nodes in the equivalent de Bruijn graph, we present the following example:

Example 2.2 *(Stationary vectors and de Bruijn graphs) Consider the de Bruijn graph depicted in Figure 2.1 in page 23, where $n_m = 2$, and $\mathcal{C} = \{0, 1\}$. From this figure we can see that, if we are at node $(0, 1)$ at time t, then we can only transit to node $(1, 0)$ or $(1, 1)$ at time $t + 1$ (cf. Table 2.1 in page 23).*

In order to describe the elements of $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$, the set of extreme points of $\mathcal{P}_{\mathcal{C}}$, we need the concept of prime cycles, whose definition is introduced below [100, pp. 678]:

Definition 2.13 *(Prime cycle) A* prime cycle *in a directed graph $\mathcal{G}_{\mathcal{V}}$ is an elementary cycle whose set of nodes do not have a proper subset which is an elementary cycle.*

The definition of prime cycle is illustrated in the next example.

Example 2.3 *(Prime cycles and de Bruijn graphs) Consider again the de Bruijn graph depicted in Figure 2.1 in page 23. According to Definition 2.13, the cycle $\{(0,1), (1,1), (1,0), (0,1)\}$ is not prime since it contains the elementary cycle $\{(0,1), (1,0), (0,1)\}$ in it. On the other hand, the elementary cycle $\{(0,1), (1,0), (0,1)\}$ is a prime cycle since it does not contain another elementary cycle.*

To introduce the next theorem, we need the definition below.

Definition 2.14 *(Descendants of an node in a de Bruijn graph) Consider a de Bruijn graph $\mathcal{G}_{\mathcal{C}^n}$. For any $x \in \mathcal{C}^n$, the set of* descendants *of x is defined as*

$$\mathcal{D}_x := \{v \in \mathcal{C}^n : (x, v) \in \mathcal{E}\}\,. \tag{2.9}$$

We have the following result [100, Theorem 6]:

Theorem 2.1 *The prime cycles of the de Bruijn graph of a Markov process of memory n_m are in one-to-one correspondence with the elements of $\mathcal{V}_{\mathcal{P}_\mathcal{C}}$, the set of extreme points of $\mathcal{P}_\mathcal{C}$. In particular, each $w_i \in \mathcal{V}_{\mathcal{P}_\mathcal{C}}$ corresponds to a uniform distribution whose support is the set of elements of a prime cycle.*

Proof *As an induction hypothesis, we assume that all measures in $\mathcal{P}_\mathcal{C}$ with less than k support points are mixtures of prime cycle measures. To start the induction, it is enough that the hypothesis is true when $k = 1$.*

Let $p \in \mathcal{P}_\mathcal{C}$ have k points in its support. By Lemma 2.1, for any $x \in \mathrm{supp}(p)$, where $\mathrm{supp}(p)$ is the support of p, we have

$$0 < p(x) \leq \sum_{v \in \mathcal{C}} p(v, x_2, \ldots, x_{n_m}) = \sum_{v \in \mathcal{C}} p(x_2, \ldots, x_{n_m}, v) = \sum_{y \in \mathcal{D}_x} p(y)\,. \quad (2.10)$$

Equation (2.10) shows that for every $x \in \mathrm{supp}(p)$, there exists a $y \in \mathrm{supp}(p)$ for which $y \in \mathcal{D}_x$. Because $\mathrm{supp}(p)$ is a finite set, this implies the existence of a cycle and hence a prime cycle in $\mathrm{supp}(p)$.

Let $a := \{a_1, \ldots, a_e\}$ be one such cycle, so that $a_i \in \mathrm{supp}(p)$ for $i \in \{1, 2\ \ldots, e\}$. Define

$$\alpha := \min_{i \in \{1,\, 2\, \ldots,\, e\}} e\, p(a_i)\,. \quad (2.11)$$

By definition $0 < \alpha \leq 1$. If $\alpha = 1$ then $p = p_a$, where p_a is a measure assigning equal probability to the elements in a, and the induction is over. If $\alpha < 1$, define the measure

$$p' := \frac{p - \alpha\, p_a}{1 - \alpha}\,. \quad (2.12)$$

It is possible to verify that p' is a probability measure. Moreover, p' has at most $k - 1$ support points, because

$$(1 - \alpha)p'(a_i) = p(a_i) - \alpha\, p_a(a_i) = p(a_i) - \min_{i \in \{1,\, 2\, \ldots,\, e\}} e\, p(a_i)\, \frac{1}{e}\,, \quad (2.13)$$

which implies that $p'(a_i) = 0$ for some $i \in \{1, 2\ \ldots, e\}$. Finally, $p' \in \mathcal{P}_\mathcal{C}$ because p' is a linear combination of the stationary measures p, and p_a.

By the induction hypothesis, p' is a mixture of prime cycle measures, and

$$p = \alpha\, p_a + (1 - \alpha)\, p'\,. \quad (2.14)$$

This shows that any $p \in \mathcal{P}_\mathcal{C}$ is a mixture of prime cycle measures. It only remains to show that all prime cycle measures are extreme points.

Let p_a be a prime cycle measure. If p_a is a mixture of stationary measures, by what has just been shown, p_a is a mixture of prime cycle measures. For any p_b in that mixture, defined for a prime cycle b, we have $\mathrm{supp}(p_b) \subset \mathrm{supp}(p_a)$. But a is prime cycle, so the only cycle contained in a is itself, and hence $b = a$. This shows that p_a is an extreme point. ■

Theorem 2.1 says that we can describe all the elements in $\mathcal{V}_{\mathcal{P}_\mathcal{C}}$ by finding all the prime cycles associated with the de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m}}$ drawn from $\mathcal{C}^{n_m}$. To find all the prime cycles in $\mathcal{G}_{\mathcal{C}^{n_m}}$, the de Bruijn graph with vertices in $\mathcal{C}^{n_m}$, we use the following lemma.

Lemma 2.2 *All the prime cycles associated with $\mathcal{G}_{\mathcal{C}^{n_m}}$ can be derived from the elementary cycles of $\mathcal{G}_{\mathcal{C}^{n_m-1}}$.*

Proof *For $x, y \in \mathcal{C}^{n_m-1}$ such that*

$$(x_2, \ldots, x_{n_m-1}) = (y_1, \ldots, y_{n_m-2}), \tag{2.15}$$

define $< x, y > \in \mathcal{C}^{n_m}$ by

$$< x, y > := (x_1, x_2, \ldots, x_{n_m-1}, y_{n_m-1}) = (x_1, y_1, \ldots, y_{n_m-2}, y_{n_m-1}). \tag{2.16}$$

For a cycle $a := \{a_1, \ldots, a_e\}$, with $a_i \in \mathcal{C}^{n_m-1}$ for all $i \in \{1, \ldots, e\}$, we create a cycle a' with elements in $\mathcal{C}^{n_m}$ by defining

$$a' := \{< a_1, a_2 >, < a_2, a_3 >, \ldots, < a_e, a_1 >\}. \tag{2.17}$$

Finally, we have that a' in (2.17) is a prime cycle if and only if:

- $< a_j, a_{j+1} > \in \mathcal{D}_{<a_i, a_{i+1}>} \Leftrightarrow i + 1 = j$,
- $a_{i+1} = a_j \Leftrightarrow i + 1 = j$,
- *a is an elementary cycle.*

Therefore, all the prime cycles in de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m}}$ can be found by computing the elementary cycles in the de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m-1}}$. ■

Lemma 2.2 states that finding all the prime cycles in $\mathcal{G}_{\mathcal{C}^{n_m}}$ is equivalent to finding all the elementary cycles in $\mathcal{G}_{\mathcal{C}^{n_m-1}}$, which can be determined using standard graph algorithms (for the examples in the next chapters, we have used the algorithm presented in [46, pp. 79–80] complemented with the one proposed in [84, pp. 157]. Appendix A presents the pseudo-codes of the algorithms employed in this thesis). To illustrate this procedure, consider the graph depicted in Figure 2.2. One elementary cycle for the graph in Figure 2.2 is given by $\{0, 1, 0\}$. Using Lemma 2.2, the elements of one prime cycle for the graph $\mathcal{G}_{\mathcal{C}^2}$ are obtained as a concatenation of the elements in the elementary cycle $\{0, 1, 0\}$. Hence, the prime cycle in $\mathcal{G}_{\mathcal{C}^2}$ associated with this elementary cycle is $\{(0, 1), (1, 0), (0, 1)\}$.

Once all the prime cycles of $\mathcal{G}_{\mathcal{C}^{n_m}}$ are found, the set $\mathcal{V}_{\mathcal{P}_\mathcal{C}}$ is fully determined. Then, for each $w_i \in \mathcal{V}_{\mathcal{P}_\mathcal{C}}$ we can generate a corresponding realization by running the corresponding prime cycle. This property will be useful in the input design method discussed in the next chapters, where numerical approximations are needed for expressions depending on the probability measure of each prime cycle.

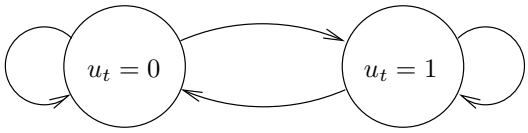

Figure 2.2: The de Bruijn graph derived from $\mathcal{C}^{n_m}$, with $n_m = 1$, and $\mathcal{C} = \{0, 1\}$.

Example 2.4 *(Generation of a sequence from a prime cycle) Consider the de Bruijn graph depicted in Figure 2.1 in page 23. From Example 2.3, we know that one prime cycle for this graph is given by* $\{(0, 1), (1, 0), (0, 1)\}$. *Therefore, a realization* $\{u_t\}_{t=1}^{N_{\text{sim}}}$ *associated with this prime cycle is computed by taking the last element of each node, i.e.,* $\{u_t\}_{t=1}^{N_{\text{sim}}} = \{1, 0, 1, 0, \ldots, ((-1)^{N_{\text{sim}}-1} + 1)/2\}$. *We notice that there is some degree of freedom on choosing from which node to start. In this example we start at node* $(0, 1)$ *at time* $t = 1$.

Remark 2.5 *The computational cost associated with the approach discussed in this thesis is mostly dominated by the effort required to compute the elementary cycles, and the Fisher information matrix for these cycles. A time bound for the computation of elementary cycles for the method presented in Appendix A is given by* $\mathcal{O}(n_{\mathcal{C}}^{n_m}(n_{\mathcal{C}} + 1)(c_e + 1))$, *where* c_e *is the number of elementary cycles given by [46, p. 77]*

$$c_e := n_{\mathcal{C}} + \sum_{i=1}^{n_{\mathcal{C}}^{n_m-1}-1} \binom{n_{\mathcal{C}}^{n_m-1}}{n_{\mathcal{C}}^{n_m-1} - i + 1} \left(n_{\mathcal{C}}^{n_m-1} - i\right)! \,. \tag{2.18}$$

For example, for a ternary signal ($n_{\mathcal{C}} = 3$), *and memory length* $n_m = 2$, *we have* $c_e = 8$, *and a time bound of order* $\mathcal{O}(324)$.

2.3 Generation of stationary sequences

In Section 2.2, we discussed a parametrization of $\mathcal{P}_{\mathcal{C}}$ to obtain a computationally tractable description of the pmfs associated with stationary vectors $u_{1:n_m}$. However, a remaining issue must be addressed: given a pmf $p \in \mathcal{P}_{\mathcal{C}}$, we want to obtain an input vector $u_{1:n_{\text{seq}}}$, where each u_t is sampled from p, for $t \in \{1, \ldots, n_{\text{seq}}\}$.

In this section we develop a procedure to generate an input sequence $u_{1:n_{\text{seq}}}$ from a given pmf $p(u_{1:n_m})$. To this end, notice that we can associate $\mathcal{G}_{\mathcal{C}^{n_m}}$ with the discrete-time Markov chain [21]

$$\pi_{k+1} = A\,\pi_k \,, \tag{2.19}$$

where $A \in \mathbb{R}^{\mathcal{C}^{n_m} \times \mathcal{C}^{n_m}}$ is a transition probability matrix[1], and $\pi_k \in \mathbb{R}^{\mathcal{C}^{n_m}}$ is the

[1]Given a set X with finite cardinality, we denote by $\mathbb{R}^{X \times X}$ the matrices with real entries, with dimensions given by the cardinality of X.

state vector.[2] In this case, there is a one-to-one correspondence between each entry of $\pi_k \in \mathbb{R}^{\mathcal{C}^{n_m}}$ and an element of $\mathcal{C}^{n_m}$.

Based on this association, $p(u_{1:n_m})$ corresponds to the stationary distribution of a Markov chain (2.19), defined as $\Pi^s \in \mathbb{R}^{\mathcal{C}^{n_m}}$. Therefore, in order to generate an input sequence $u_{1:n_{\text{seq}}}$ from $p(u_{1:n_m})$, we can design a Markov chain having $p(u_{1:n_m})$ as its stationary distribution, and simulate this Markov chain to generate $u_{1:n_{\text{seq}}}$ from its samples.

To continue, we denote by $A_{rl} \in \mathbb{R}$ the (r,l)-entry of A. We notice that the indices of A are not numerical, but belong to $\mathcal{C}^{n_m}$. Based on the previous notation, a valid A for the Markov chain (2.19) must satisfy

$$A_{rl} \geq 0\,, \text{ for all } r, l \in \mathcal{C}^{n_m}\,, \tag{2.20}$$

$$\sum_{r \in \mathcal{C}^{n_m}} A_{rl} = 1\,, \text{ for all } l \in \mathcal{C}^{n_m}\,, \tag{2.21}$$

$$A_{rl} = 0\,, \text{ if } (l,\, r) \notin \mathcal{E}\,. \tag{2.22}$$

It can be proven that a matrix A satisfying (2.20)-(2.21) has 1 as an eigenvalue [43]. Furthermore, if the Markov chain is ergodic, the unique eigenvector $\Pi^s \in \mathbb{R}^{\mathcal{C}^{n_m}}$ associated with this eigenvalue is the unique stationary pmf of $\mathcal{C}^{n_m}$ (up to a scaling factor), satisfying

$$\Pi^s = A\Pi^s\,. \tag{2.23}$$

The task is to design a transition probability matrix satisfying (2.20)-(2.23). There is an extensive literature on how to optimize the mixing time of the resulting Markov chain (i.e., the time required to obtain samples distributed according to the stationary measure of the Markov chain, see, e.g., [4, 36] and the references therein). However, these works assume that the graph is undirected or reversible, which implies that A must have a particular structure (e.g., A a symmetric matrix). Since the structure of the graph $\mathcal{G}_{\mathcal{C}^{n_m}}$ does not satisfy in general the required properties, most existing methods cannot be applied here.

Below, we develop a method to design a transition probability matrix for the de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m}}$. The idea is that if we parameterize the transition probabilities of a Markov chain of memory n in terms of the stationary probabilities of a Markov chain of memory $n+1$, we obtain a computationally tractable description of $\mathcal{P}_{\mathcal{C}}$, as discussed in Section 2.2. Given the optimal stationary probabilities, the proposed algorithm gives a unique mapping between the optimized stationary probabilities and the transition matrix with memory n using that

$$\mathbf{P}\{u_t | u_{t-1},\, \ldots,\, u_{t-n}\} = \frac{\mathbf{P}\{(u_t,\, \ldots,\, u_{t-n})\}}{\sum_{u_t \in \mathcal{C}} \mathbf{P}\{(u_t,\, \ldots,\, u_{t-n})\}}\,, \tag{2.24}$$

where $\mathbf{P}$ denotes the stationary probability measure of the Markov chain (i.e., defined by the entries of Π^s), and $\mathbf{P}\{\cdot|\cdot\}$ denotes the conditional probability measure.

[2]Note that equation (2.19) is not in standard Markov chain notation (defined as the transpose of (2.19)).

Algorithm 2.1 Design of a transition probability matrix

INPUTS: A pmf p, its memory n_m, and the alphabet $\mathcal{C}$.
OUTPUT: A transition probability matrix A with stationary distribution p.

1: For each $r \in \mathcal{C}^{n_m}$, define

$$\mathcal{A}_r := \{l \in \mathcal{C}^{n_m} : (l, r) \in \mathcal{E}\}. \tag{2.26}$$

In other words, $\mathcal{A}_r$ is the set of ancestors of r.
2: For each r, $l \in \mathcal{C}^{n_m}$, let

$$A_{rl} = \begin{cases} \frac{\mathbf{P}\{r\}}{\sum\limits_{k \in \mathcal{A}_r} \mathbf{P}\{k\}}, & \text{if } l \in \mathcal{A}_r \text{ and } \sum\limits_{k \in \mathcal{A}_r} \mathbf{P}\{k\} \neq 0, \\ \frac{1}{\#\mathcal{A}_r}, & \text{if } l \in \mathcal{A}_r \text{ and } \sum\limits_{k \in \mathcal{A}_r} \mathbf{P}\{k\} = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2.27}$$

To continue, we need the following result:

Lemma 2.3 *The stationary probability measure* $\mathbf{P}$*, corresponding to* $p(u_{1:n_m})$*, satisfies*

$$\sum_{r=1}^{c_{\text{seq}}} \mathbf{P}\{(v_1, \ldots, v_{n_m-1}, s_r)\} = \sum_{r=1}^{c_{\text{seq}}} \mathbf{P}\{(s_r, v_1, \ldots, v_{n_m-1})\}, \tag{2.25}$$

for all $(v_1, \ldots, v_{n_m-1}) \in \mathcal{C}^{n_m-1}$.

Lemma 2.3 follows since $p(u_{1:n_m})$ is the projection of a stationary distribution, cf. Lemma 2.1. Based on this fact, we can design a transition probability matrix A for $\mathcal{G}_{\mathcal{C}^{n_m}}$ as described in Algorithm 2.1.

Algorithm 2.1 introduces a method to design valid transition probability matrices when Π^s satisfies Lemma 2.3. For an element $r \in \mathcal{C}^{n_m}$ with nonzero probability, Algorithm 2.1 assigns nonzero transition probability only to those elements in the set of ancestors of r. If an element $v \in \mathcal{C}^{n_m}$ has zero probability, the algorithm assigns equal probability to those elements in the set of ancestors of v.

The next theorem establishes the correctness of the algorithm.

Theorem 2.2 *Given a stationary probability measure* $\mathbf{P}$ *satisfying Lemma 2.3, then the matrix* $A \in \mathbb{R}^{\mathcal{C}^{n_m} \times \mathcal{C}^{n_m}}$ *designed by Algorithm 2.1 is a transition probability matrix satisfying* (2.20)-(2.23).

Proof *Properties* (2.20) *and* (2.22) *are trivially satisfied by the construction of* A*. To establish* (2.21) *and* (2.23)*, we need to analyze the structure of the transition probability matrix* A *associated with a de Bruijn graph. From the definition of* $\mathcal{E}$

(cf. (2.2)*), we have that*

$$\sum_{l\in\mathcal{A}_r}\mathbf{P}\{l\}=\sum_{l=1}^{n_{\mathcal{C}}}\mathbf{P}\{(s_l,\,r(1),\,\ldots,\,r(n_m-1))\}\,. \tag{2.28}$$

To proceed, we need the set of descendants of r*, denoted by* $\mathcal{D}_r$*. From the definition of* $\mathcal{E}$ *in equation* (2.2)*, we have that* $\#\mathcal{D}_r=\#\mathcal{A}_r=n_{\mathcal{C}}$*.*

First, we prove (2.21)*. Consider first an* $l\in\mathcal{C}^{n_m}$ *such that* $\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}\neq 0$ *for all* $r\in\mathcal{D}_l$*. Then,*

$$\mathcal{D}_l=\{(l(2),\,\ldots,\,l(n_m),\,s_1),\,\ldots,\,(l(2),\,\ldots,\,l(n_m),\,s_{c_{\rm seq}})\}\,. \tag{2.29}$$

In addition, for any $r\in\mathcal{D}_l$*,*

$$\mathcal{A}_r=\{(s_1,\,l(2),\,\ldots,\,l(n_m)),\,\ldots,\,(s_{c_{\rm seq}}\,,l(2),\,\ldots,\,l(n_m))\}\,. \tag{2.30}$$

Equation (2.30) *shows that the sets* $\mathcal{A}_r$ *are equal for all* $r\in\mathcal{D}_l$*. Therefore, the sums* $\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}$ *are equal (and nonzero) for all* $r\in\mathcal{D}_l$*, hence*

$$\sum_{r\in\mathcal{C}^{n_m}}A_{rl}=\sum_{r\in\mathcal{D}_l}\frac{\mathbf{P}\{r\}}{\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}}=\frac{\sum_{r\in\mathcal{D}_l}\mathbf{P}\{r\}}{\sum_{k\in\mathcal{A}_{\tilde r}}\mathbf{P}\{k\}}\,, \tag{2.31}$$

for any fixed $\tilde r\in\mathcal{D}_l$*. Furthermore, in the light of* (2.29)-(2.30)*, we can rewrite* (2.31) *as*

$$\sum_{r\in\mathcal{C}^{n_m}}A_{rl}=\frac{\sum_{r=1}^{n_{\mathcal{C}}}\mathbf{P}\{(l(2),\,\ldots,\,l(n_m),\,s_r)\}}{\sum_{k=1}^{n_{\mathcal{C}}}\mathbf{P}\{s_k,\,l(2),\,\ldots,\,l(n_m)\}}=1, \tag{2.32}$$

where the last equality follows from Lemma 2.3.

On the other hand, if $l\in\mathcal{C}^{n_m}$ *is such that* $\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}=0$ *for all* $r\in\mathcal{C}^{n_m}$*, we write*

$$\sum_{r\in\mathcal{C}^{n_m}}A_{rl}=\sum_{r\in\mathcal{D}_l}\frac{1}{\#\mathcal{A}_r}=\sum_{r\in\mathcal{D}_l}\frac{1}{\#\mathcal{D}_l}=1\,. \tag{2.33}$$

The results presented in (2.32)-(2.33) *establish* (2.21)*.*

Now we prove (2.23)*. For each* $r\in\mathcal{C}^{n_m}$ *such that* $\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}\neq 0$*, we have that the* r*-th element of the product* $A\Pi^s$ *(denoted by* π_r^s*) is given by*

$$\pi_r^s=\frac{\mathbf{P}\{r\}}{\sum_{l\in\mathcal{A}_r}\mathbf{P}\{l\}}\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}=\mathbf{P}\{r\}\,. \tag{2.34}$$

On the other hand, for each $r\in\mathcal{C}^{n_m}$ *such that* $\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}=0$*, we can consider an* $l\in\mathcal{C}^{n_m}$ *such that* $r\in\mathcal{D}_l$*. According to* (2.29)*,* (2.30)*, and using Lemma 2.3, we can conclude that*

$$\sum_{\tilde r\in\mathcal{D}_l}\mathbf{P}\{\tilde r\}=\sum_{k\in\mathcal{A}_r}\mathbf{P}\{k\}=0\,, \tag{2.35}$$

which implies that

$$\mathbf{P}\{k\} = 0 \tag{2.36}$$

for all $k \in \mathcal{D}_l$*, and in particular, for* $k = r$*. Since* $l \in \mathcal{C}^{n_m}$ *is arbitrary,* (2.36) *is true for all* $l \in \mathcal{C}^{n_m}$ *such that* $r \in \mathcal{D}_l$*. Hence,* (2.34) *is also satisfied for each* $r \in \mathcal{C}^{n_m}$ *such that* $\sum_{k \in \mathcal{A}_r} \mathbf{P}\{k\} = 0$*, which establishes* (2.23)*. This concludes the proof.* ■

The transition probability matrix designed using Algorithm 2.1 has the following property:

Theorem 2.3 *The transition probability matrix* $A \in \mathbb{R}^{\mathcal{C}^{n_m} \times \mathcal{C}^{n_m}}$ *designed by Algorithm 2.1 has all its eigenvalues in the region* $\mathbf{D} := \{z \in \mathbb{C} : |z| \leq 1\}$*. In addition,* A *has at most* $n_{\mathcal{C}}^{n_m - 1}$ *nonzero eigenvalues in* $\mathbf{D}$*.*

Proof *The first statement follows since* A *is a transition probability matrix [43], according to Theorem 2.2.*

To establish the second statement, notice that, from (2.29)-(2.30)*, for each* $l \in \mathcal{C}^{n_m}$ *we have that* $\mathcal{A}_r$ *is the same for all* $r \in \mathcal{D}_l$*, which means that the columns of* A *can be partitioned into* $n_{\mathcal{C}}^{n_m - 1}$ *groups of* $n_{\mathcal{C}}$ *identical columns. Therefore, the number of nonzero eigenvalues of* A *in* $\mathbf{D}$ *is at most* $n_{\mathcal{C}}^{n_m - 1}$*, since there are at most* $n_{\mathcal{C}}^{n_m - 1}$ *linear independent columns in* A*. This concludes the proof.* ■

Remark 2.6 *There are, in general, several transition matrices having a given* $\mathbf{P}$ *as stationary probability measure, subject to a graph constraint (a prescribed set of edges). Algorithm 2.1 provides only one such choice based on the constraint that* A *corresponds to a Markov chain of order* n *instead of* $n+1$*. Among those transition matrices, it would be preferable to select the one with the fastest mixing time, i.e., for which the Markov chain reaches the stationary distribution as quickly as possible. The most common criterion to define mixing time is the second largest eigenvalue modulus (SLEM). A Monte Carlo study, for* $n_{\mathcal{C}} = 2$ *and* $n_m = 2$*, based on uniform sampling from the set of transition matrices giving a specific* $\mathbf{P}$ *(which can be shown to be a polytope) has empirically shown that the* A *matrix given by Algorithm 2.1 is within the* $7\,\%$ *with lowest SLEM, which suggests that Algorithm 2.1 gives a reasonable (but improvable) mixing time. One way to further reduce the SLEM of* A *is by performing gradient descent over the transition probabilities in* A*, starting from the matrix designed in Algorithm 2.1. Another option to reduce the SLEM is by exploiting the full memory of the Markov chain when designing the transition probability matrix; this is part of the future work on the subject.*

Remark 2.7 *For simplicity, the results introduced in this thesis are discussed for scalar sequences. However, an immediate extension of this technique to the case of sequences of vectors can be done. In the case of sequences of vectors with* n_u *entries, the states associated with each node in the de Bruijn graph are the possible values of an* $n_u \times n_m$ *matrix, where the* i*-th row describes the feasible states for*

the stationary process in the i-th entry. With this modification, the method can be directly employed to solve input design for MIMO models.

2.4 Conclusion

In this chapter, we introduced the elements from graph theory and stationary processes employed in the thesis. We presented how de Bruijn graphs can be employed to describe the set of stationary processes with finite memory. In particular, it was shown that the set of stationary processes with finite memory and finite alphabet can be described as the convex hull of the probability measures associated with the prime cycles of the equivalent de Bruijn graph.

We are interested in obtaining realizations from a given pmf in the set of stationary processes. To obtain a realization with the desired distribution, we consider the samples as the output of a Markov chain with stationary distribution given by the desired pmf. To obtain the associated transition probability matrix, this chapter introduced an algorithm to build the transition probabilities resulting in a Markov chain with the desired pmf as stationary distribution. However, the algorithm does not guarantee that the resulting transition probability matrix has the optimal mixing time, but it could be possible to improve this property by using numerical optimization techniques, which will be addressed in future work.

Chapter 3

Optimal input design for nonlinear output-error models

In Chapter 2, we discussed the connection between stationary processes with finite alphabet and memory, and de Bruijn graphs. The results introduced there will now be employed for developing methods designing input sequences for nonlinear models.

As mentioned in Chapter 1, the main difficulty when designing input sequences to identify nonlinear models is that frequency domain techniques cannot be employed. Therefore, most of the results in input design for linear models cannot be extended to the nonlinear case. In addition, most of the existing results on input design cannot handle amplitude constraints, which could arise due to physical or safety considerations.

This chapter introduces an approach for designing input sequences to identify nonlinear output-error (NOE) models. The method considers the design of an input sequence as a realization of a stationary process with finite memory and finite alphabet. The optimal stationary process is obtained by maximizing a scalar cost function of the Fisher information matrix. By parameterizing the set of stationary processes in terms of its extreme points, we employ de Bruijn graphs to obtain the probability measures associated with the extreme points, and then approximate the corresponding Fisher information matrix for each extreme point. Consequently, the problem becomes convex also for nonlinear models. Numerical examples show that the technique discussed in this chapter is consistent with existing results in the literature, and that it is an attractive alternative for designing input sequences for identification of nonlinear models.

The present chapter can be seen as an extension of the results in [54] and [6, 7]. The main difference with [6, 7] is that we optimize over the stationary pmf associated with the Markov chain, instead of directly optimizing over the transition probabilities. This approach results in a convex problem (which cannot be achieved in [6, 7], where optimization techniques guaranteeing local optima must

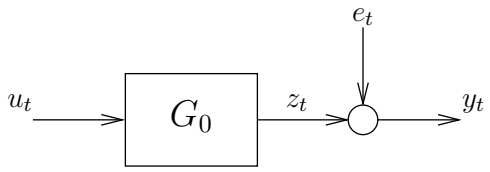

Figure 3.1: Block diagram of a (possibly nonlinear) system.

be employed). In [54] a similar approach to the one presented in this chapter is discussed, but restricted to the analysis to nonlinear FIR systems. By using the finite memory property of nonlinear FIR models, the input design problem in [54] is solved in terms of an input realization of finite length. However, the results in [54] cannot be employed to design input sequences for identification of more general nonlinear output-error models, since the models will generally depend on the entire past input sequence. In this line, the present chapter extends the analysis to more general nonlinear model structures, which includes nonlinear FIR systems.

3.1 Problem formulation

Consider the single input, single output (SISO) time invariant system depicted in Figure 3.1. Here, G_0 is a dynamical system (possibly nonlinear), defined for $t \geq 1$ as

$$G_0(u_t) := \begin{cases} x_{t+1} = f_0(x_t,\, u_t)\,, \\ z_t = h_0(x_t,\, u_t)\,, \\ x_1 = \mu\,, \end{cases} \tag{3.1}$$

where $\{e_t\}$ is white noise sequence with zero mean and finite variance λ_e, $u_t \in \mathbb{R}$ is the input, $x_t \in \mathbb{R}^{n_x}$ are the internal states of G_0 with initial condition $\mu \in \mathbb{R}^{n_x}$, and $y_t \in \mathbb{R}$ is the measured output. We assume that we have a model structure G, defined for any $\theta \in \Theta \subset \mathbb{R}^{n_\theta}$ as

$$G(u_t;\, \theta) := \begin{cases} x_{t+1} = f(x_t,\, u_t;\, \theta)\,, \\ z_t = h(x_t,\, u_t;\, \theta)\,, \\ x_1 = \mu\,. \end{cases} \tag{3.2}$$

We assume that there exists a $\theta_0 \in \Theta$ such that $G(u_t;\, \theta_0) = G_0(u_t)$ [59], i.e., there is no undermodelling. Notice that the noise, e_t, is assumed to enter only at the output. To continue, we introduce the following definition:

Definition 3.1 *Consider a bounded signal $\{u_t\}$, $|u_t| \leq K$ $(K > 0)$, and a nonlinear system $y_t = G_0(u_t)$. We say that G_0 is* exponentially stable *if and only if there are constants $C > 0$ (depending possibly on K), $0 < \delta < 1$, such that for all $t,\, s \in \mathbb{Z}$,*

$$|G_0(u_t) - G_0(u_t^s)| < C\delta^{t-s}\,, \tag{3.3}$$

where

$$u_k^s = \begin{cases} u_k, & k > s, \\ 0, & \textit{otherwise.} \end{cases} \tag{3.4}$$

We notice that Definition 3.1 differs from that given by [58] since it considers deterministic systems, and it is not defined in terms of moments of order 4.

The objective in this chapter is to design an input signal $u_{1:n_{\text{seq}}} = (u_1, \ldots, u_{n_{\text{seq}}})$ as a realization of a stationary process, such that the system (3.1) can be estimated with maximum accuracy as defined by a scalar function of the Fisher information matrix $\mathcal{I}_F^e$ [59]. $\mathcal{I}_F^e$ can be computed using Lemma 1.1 as[1]

$$\mathcal{I}_F^e = \frac{1}{\lambda_e} \mathbf{E}_e \left\{ \sum_{t=1}^{n_{\text{seq}}} \psi_t^{\theta_0}(u_t) \psi_t^{\theta_0}(u_t)^\top \middle| u_{1:n_{\text{seq}}} \right\}, \tag{3.5}$$

where

$$\psi_t^{\theta_0}(u_t) := \left. \frac{d\, \hat{y}_t(u_t,\, \theta)}{d\theta} \right|_{\theta=\theta_0}, \tag{3.6a}$$

$$\hat{y}_t(u_t,\, \theta) := G(u_t;\, \theta)\,, \tag{3.6b}$$

and $\theta,\, \theta_0 \in \Theta$. Since we are interested in computing $u_{1:n_{\text{seq}}}$ as a realization from a stationary process, we need to quantify the Fisher information matrix in terms of its expected value with respect to $u_{1:n_{\text{seq}}}$. We define the result of this expected value as the *per-sample* Fisher information matrix, which is computed as

$$\begin{aligned} \mathcal{I}_F &:= \mathbf{E}_{u_{1:n_{\text{seq}}}} \{\mathcal{I}_F^e\} \\ &= \frac{1}{\lambda_e} \mathbf{E}_{u_{1:n_{\text{seq}}},\, e} \left\{ \sum_{t=1}^{n_{\text{seq}}} \psi_t^{\theta_0}(u_t) \psi_t^{\theta_0}(u_t)^\top \right\}. \end{aligned} \tag{3.7}$$

Equation (3.6b) does not depend on the noise realization. Therefore, we can rewrite (3.7) as

$$\mathcal{I}_F = \frac{1}{\lambda_e} \int_{u_{1:n_{\text{seq}}} \in \mathbb{R}^{n_{\text{seq}}}} \sum_{t=1}^{n_{\text{seq}}} \psi_t^{\theta_0}(u_t) \psi_t^{\theta_0}(u_t)^\top \, dP(u_{1:n_{\text{seq}}})\,, \tag{3.8}$$

where $P(u_{1:n_{\text{seq}}})$ is the cdf of $u_{1:n_{\text{seq}}}$.

We note that (3.8) depends on $P(u_{1:n_{\text{seq}}})$. Therefore, the input design problem we will consider is to find a cdf, $P^{\text{opt}}(u_{n_{\text{seq}}})$, which maximizes a scalar function of (3.8). We define this function as $h\colon \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}$. As it is customary in input design [33, 45, 59], h is assumed to be a concave, nondecreasing, matrix function [5, pp. 108]. The assumption of h being nondecreasing function is to guarantee that, for

[1] In this section we add a subscript to the expected value to emphasize the process with respect to which the expectation is taken.

any two matrices X and Y in the positive semidefinite cone, we have $h(X) \geq h(Y)$ when $X \succeq Y$. Several choices of h have been proposed in the literature [74]. In this chapter, we leave the selection of h open to the user.

Since $P(u_{1:n_{\text{seq}}})$ has to be a stationary cdf, the optimization must be constrained to the set

$$\mathcal{P}_{n_{\text{seq}}} := \Bigg\{ P : \mathbb{R}^{n_{\text{seq}}} \to \mathbb{R} |\, P(\mathbf{x}) \geq 0,\, \forall \mathbf{x} \in \mathbb{R}^{n_{\text{seq}}};$$
$$P \text{ is monotone nondecreasing};$$
$$\lim_{x_i \to \infty} P(x_1, \ldots, x_{n_{\text{seq}}}) = 1, \quad \text{for } i = 1, \ldots, n_{\text{seq}}$$
$$\int_{v \in \mathbb{R}} dP\left([v,\, \mathbf{z}]\right) = \int_{v \in \mathbb{R}} dP\left([\mathbf{z},\, v]\right),\, \forall \mathbf{z} \in \mathbb{R}^{n_{\text{seq}}-1} \Bigg\}. \quad (3.9)$$

As mentioned in Chapter 2, the last condition in (3.9) guarantees that $P \in \mathcal{P}_{n_{\text{seq}}}$ is the projection of the cdf of a stationary sequence [100].

Based on Definition 3.1, we impose the following assumption over $\psi_t^{\theta_0}(u_t)\psi_t^{\theta_0}(u_t)^\top$:

Assumption 3.1 *$\psi_t^{\theta_0}(u_t)\psi_t^{\theta_0}(u_t)^\top$ is exponentially stable with constants $C_\psi > 0$, $0 < \delta_\psi < 1$, where $\psi_t^{\theta_0}(u_t)$ defined in* (3.6).

Remark 3.1 *Assumption 3.1 is satisfied for the systems considered in this chapter, since the system in equation* (3.1) *is required to be stable in order to design an input sequence.*

The input design problem can be summarized as:

Problem 3.1 *(Input design for nonlinear output-error models) Design an optimal input signal $u_{1:n_{\text{seq}}}^{\text{opt}} \in \mathbb{R}^{n_{\text{seq}}}$ as a realization from the cdf $P^{\text{opt}}(u_{1:n_{\text{seq}}})$, given by*

$$P^{\text{opt}}(u_{1:n_{\text{seq}}}) = \arg \max_{P \in \mathcal{P}_{n_{\text{seq}}}} h(\mathcal{I}_F(P)), \quad (3.10)$$

where $h\colon \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}$ is a concave, nondecreasing, matrix function,

$$\mathcal{I}_F(P) = \frac{1}{\lambda_e} \int_{u_{1:n_{\text{seq}}} \in \mathbb{R}^{n_{\text{seq}}}} \sum_{t=1}^{n_{\text{seq}}} \psi_t^{\theta_0}(u_t)\psi_t^{\theta_0}(u_t)^\top \, dP(u_{1:n_{\text{seq}}}), \quad (3.11)$$

and where $\psi_t^{\theta_0}(u_t) \in \mathbb{R}^{n_\theta}$ is defined as in (3.6)*, satisfying Assumption 3.1.*

Problem 3.1 is difficult to solve. Indeed, since n_{seq} can be large, the input design problem potentially involves dealing with a very high dimensional integral (3.8), which can be computationally intractable. To address this issue, we can restrict the input signal u_t to be a stationary process of finite memory, i.e., u_t can be

assumed to be a Markov process of order n_m. This means that $P(u_{1:n_{\text{seq}}})$ can be completely described by the n_m-dimensional projection $P(u_{1:n_m})$ [100].

As it was discussed for $P(u_{1:n_{\text{seq}}})$, $P(u_{1:n_m})$ is also restricted to be the projection of a stationary cdf. Therefore, the optimization must be constrained to the set

$$\mathcal{P} = \Bigg\{ P : \mathbb{R}^{n_m} \to \mathbb{R} |\, P(\mathbf{x}) \geq 0,\, \forall \mathbf{x} \in \mathbb{R}^{n_m}; \\ P \text{ is monotone nondecreasing ;} \\ \lim_{x_i \to \infty} P(x_1, \ldots, x_{n_m}) = 1, \quad \text{for } i = 1, \ldots, n_m \\ \int_{v \in \mathbb{R}} dP\left([v,\, \mathbf{z}]\right) = \int_{v \in \mathbb{R}} dP\left([\mathbf{z},\, v]\right) ,\forall \mathbf{z} \in \mathbb{R}^{n_m - 1} \Bigg\} . \quad (3.12)$$

For computational tractability, we further constrain u_t to belong to a finite alphabet $\mathcal{C}$ with cardinality $n_{\mathcal{C}}$. With this assumption, it is convenient to work with the pmf, $p(u_{1:n_m})$ (cf. Definition 2.10), rather than the cdf $P(u_{1:n_m})$. In addition, we can define the constraint set of $p(u_{1:n_m})$ as:

$$\mathcal{P}_{\mathcal{C}} = \{ p : \mathcal{C}^{n_m} \to \mathbb{R} |\, p(\mathbf{x}) \geq 0,\, \forall \mathbf{x} \in \mathcal{C}^{n_m}; \\ \sum_{\mathbf{x} \in \mathcal{C}^{n_m}} p(\mathbf{x}) = 1; \\ \sum_{v \in \mathcal{C}} p\left([v,\, \mathbf{z}]\right) = \sum_{v \in \mathcal{C}} p\left([\mathbf{z},\, v]\right) ,\forall \mathbf{z} \in \mathcal{C}^{n_m - 1} \Bigg\} . \quad (3.13)$$

Finally, the relaxed input design problem can be summarized as:

Problem 3.2 *(Input design for nonlinear output-error models, relaxed problem) Design an optimal input signal $u^{\text{opt}}_{1:n_{seq}} \in \mathcal{C}^{n_{\text{seq}}}$ as a realization from the projected pmf $p^{\text{opt}}(u_{1:n_m})$, given by*

$$p^{\text{opt}}(u_{1:n_m}) = \arg\max_{p \in \mathcal{P}_{\mathcal{C}}} h(\mathcal{I}_F(p)) \,, \quad (3.14)$$

where $h\colon \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}$ is a concave, nondecreasing, matrix function,

$$\mathcal{I}_F(p) = \frac{1}{\lambda_e} \sum_{u_{1:n_{\text{seq}}} \in \mathcal{C}^{n_{\text{seq}}}} \sum_{t=1}^{n_{\text{seq}}} \psi_t^{\theta_0}(u_t) \psi_t^{\theta_0}(u_t)^\top \, p(u_{1:n_{\text{seq}}}) \,, \quad (3.15)$$

and where $\psi_t^{\theta_0}(u_t) \in \mathbb{R}^{n_\theta}$ is defined as in (3.6)*, satisfying Assumption 3.1.*

Remark 3.2 *Problem 3.2 assumes the existence of prior information regarding the model parameter $\theta \in \Theta$. In this chapter we assume that there exists a prior estimate $\hat{\theta}_N$ that can be used to solve Problem 3.2. This difficulty can be overcome by implementing a robust experiment design scheme [74] or via an adaptive procedure,*

where the input signal is redesigned as more information is being collected from the system [29, 73]. This issue goes beyond the scope of this thesis, and will be addressed in a future work.

An approach to solve Problem 3.2 is discussed in the next section.

3.2 Input design via graph theory

To solve Problem 3.2, we need to parameterize the set $\mathcal{P}_{\mathcal{C}}$ in a computationally tractable manner. To this end, we will use the results introduced in Section 2.2 to describe any element in $\mathcal{P}_{\mathcal{C}}$ as a convex combination of its extreme points. In this section we follow the notation introduced in Section 2.2, and we denote by $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$ the set of extreme points of $\mathcal{P}_{\mathcal{C}}$.

By Theorem 2.1 and Lemma 2.2, we know how to compute the probability measures associated with the elements In $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$. Indeed, Theorem 2.1 states that the prime cycles in the de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m}}$ are in one-to-one correspondence with the elements in $\mathcal{V}_{\mathcal{P}}$ (the set of extreme points of $\mathcal{C}^{n_m}$), with uniform distribution whose support are the elements in the prime cycle. Furthermore, Lemma 2.2 states that the prime cycles in the de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m}}$ are derived from the elementary cycles in the de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m-1}}$. Therefore, we can define the Fisher information matrix corresponding to the i-th prime cycle and element $w_i \in \mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$ as

$$\mathcal{I}_F^{(i)} := \frac{1}{\lambda_e} \sum_{u_{1:n_m} \in \mathcal{C}^{n_m}} \sum_{t=1}^{n_m} \psi_t^{\theta_0}(u_t^i) \psi_t^{\theta_0}(u_t^i)^\top \, w_i(u_{1:n_m}) \,, \tag{3.16}$$

for all $i \in \{1, \ldots, n_{\mathcal{V}}\}$. Notice that each $\mathcal{I}_F^{(i)}$ is associated with the i-th prime cycle. The explicit computation of $\mathcal{I}_F^{(i)}$ for the nonlinear model (3.2) is often intractable, since it requires the computation of expected values of nonlinear functions of u_t^i. Therefore, a numerical approximation of (3.16) is needed. To this end, instead of approximating $\mathcal{I}_F^{(i)}$ as an average over different realizations of the input sequence $u_{1:n_{\text{seq}}}$, we consider an approximation of $\mathcal{I}_F^{(i)}$ as an average over one realization of length N_{sim}, which can be written as

$$\mathcal{I}_F^{(i)} \approx \frac{1}{\lambda_e \, N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \psi_t^{\theta_0}(u_t^i) \psi_t^{\theta_0}(u_t^i)^\top \,, \tag{3.17}$$

where N_{sim} is sufficiently large. The approximation (3.17) converges to $\mathcal{I}_F^{(i)}$ as $N_{\text{sim}} \to \infty$ since $\psi_t^{\theta_0}(u_t) \psi_t^{\theta_0}(u_t)^\top$ satisfies Assumption 3.1 (see Appendix B for a proof of this statement). The approximation error incurred when (3.17) is employed to compute $\mathcal{I}_F^{(i)}$ is of order $\delta_\psi^{N_{\text{sim}}}$.

To illustrate how to generate $\{u_t^i\}_{t=1}^{N_{\text{sim}}}$ for a particular w_i, we recall Example 2.4 in page 30, and the graph depicted in Figure 2.1. One prime cycle for that graph

is given by $\{(0, 1), (1, 0), (0, 1)\}$. Therefore, the sequence $\{u_t^i\}_{t=1}^{N_{\rm sim}}$ is obtained by taking the last element of each node, i.e.,

$$\{u_t^i\}_{t=1}^{N_{\rm sim}} = \{1, 0, 1, 0, \ldots, ((-1)^{N_{\rm sim}+1} + 1)/2\}\,.$$

Once the approximation (3.17) is made for all the elements in $\mathcal{V}_{\mathcal{P}_\mathcal{C}}$, we can compute an approximation of the information matrix, $\mathcal{I}_F(p)$, associated with the elements in $\mathcal{P}_\mathcal{C}$ as a convex combination of the $\mathcal{I}_F^{(i)}$'s, $i \in \{1, \ldots, n_\mathcal{V}\}$. If we define $\alpha := \{\alpha_1, \ldots, \alpha_{n_\mathcal{V}}\} \in \mathbb{R}^{n_\mathcal{V}}$, we can write

$$\mathcal{I}_F^{\rm app}(\gamma) := \sum_{i=1}^{n_\mathcal{V}} \alpha_i \mathcal{I}_F^{(i)}\,, \tag{3.18a}$$

$$\sum_{i=1}^{n_\mathcal{V}} \alpha_i = 1\,, \tag{3.18b}$$

$$\alpha_i \geq 0\,,\ \text{for all } i \in \{1, \ldots, n_\mathcal{V}\}\,, \tag{3.18c}$$

where $\mathcal{I}_F^{\rm app}(\gamma)$ is the approximation of the information matrix $\mathcal{I}_F(p)$ associated with the elements of $\mathcal{P}_\mathcal{C}$.

To summarize, the proposed method for the design of input signals in $\mathcal{C}^{n_m}$ can be described as follows:

1. Compute all the elementary cycles of $\mathcal{G}_{\mathcal{C}^{(n_m-1)}}$ using Algorithm A.2-A.3 in Appendix A.
2. Compute all the prime cycles of $\mathcal{G}_{\mathcal{C}^{n_m}}$ from the elementary cycles of $\mathcal{G}_{\mathcal{C}^{n_m-1}}$ as explained in Lemma 2.2 in page 29.
3. Generate the input signals $\{u_t^i\}_{t=1}^{N_{\rm sim}}$ from the prime cycles of $\mathcal{G}_{\mathcal{C}^{n_m}}$, for each $i \in \{1, \ldots, n_\mathcal{V}\}$.
4. For each $i \in \{1, \ldots, n_\mathcal{V}\}$, approximate $\mathcal{I}_F^{(i)}$ using (3.17).
5. Define $\alpha = \{\alpha_1, \ldots, \alpha_{n_\mathcal{V}}\} \in \mathbb{R}^{n_\mathcal{V}}$. Find $\alpha^{\rm opt} := \{\alpha_1^{\rm opt}, \ldots, \alpha_{n_\mathcal{V}}^{\rm opt}\}$ by solving the following approximation of Problem 3.2:

$$\alpha^{\rm opt} = \arg \max_{\alpha \in \mathbb{R}^{n_\mathcal{V}}} h(\mathcal{I}_F^{\rm app}(\alpha))\,, \tag{3.19}$$

where $h : \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}$ is a concave, nondecreasing matrix function,

$$\mathcal{I}_F^{\rm app}(\gamma) = \sum_{i=1}^{n_\mathcal{V}} \alpha_i \mathcal{I}_F^{(i)}\,, \tag{3.20a}$$

$$\sum_{i=1}^{n_\mathcal{V}} \alpha_i = 1\,, \tag{3.20b}$$

$$\alpha_i \geq 0\,,\ \text{for all } i \in \{1, \ldots, n_\mathcal{V}\}\,, \tag{3.20c}$$

and $\mathcal{I}_F^{(i)}$ is given by (3.17).

6. Compute the optimal pmf p^{opt} as

$$p^{\text{opt}} = \sum_{i=1}^{n_{\mathcal{V}}} \alpha_i^{\text{opt}} \, w_i \, . \tag{3.21}$$

7. Generate $u_{1:n_{\text{seq}}}^{\text{opt}}$ using Algorithm 2.1 in page 32, with p^{opt} as stationary distribution of the Markov chain.

The procedure mentioned above computes $u_{1:n_{\text{seq}}}^{\text{opt}}$ as a realization from the optimal pmf, $p^{\text{opt}}(u_{1:n_m})$, obtained by solving (3.19). Notice that $\mathcal{I}_F^{\text{app}}(\gamma)$ in (3.20a) is linear in the decision variables. Therefore, problem (3.19)-(3.20) is convex.

Remark 3.3 *Steps (1) to (3) are independent of the system for which the input is designed. Therefore, once the steps (1) to (3) are computed, they can be re-used for input design in different model structures and systems.*

Remark 3.4 *The approximate solution to Problem 3.2 given by* (3.19) *may be nonunique. In that case,* (3.19) *will return the weighting vector associated with one optimal pmf. Moreover, even if the optimal pmf $p^{\text{opt}}(u_{1:n_m})$ is unique, the optimal input sequence $u_{1:n_{\text{seq}}}$ is not. Indeed, $u_{1:n_{\text{seq}}}$ is a realization of a Markov chain with stationary distribution $p^{\text{opt}}(u_{1:n_m})$.*

3.3 Reducible Markov chains

When the optimization (3.19) is solved, it might occur that the resulting p^{opt} is associated to a reducible Markov chain. The last means that there exists at least two sets of vertices in the Markov chain, such that each set of vertices cannot be accessed from the others. This is an issue of the proposed approach, since Algorithm 2.1 cannot be employed to generate samples from p^{opt} if it is the pmf of a reducible Markov chain.

One possibility to overcome this issue is to perturb the optimal pmf p^{opt} in order to achieve an irreducible Markov chain, and then use Algorithm 2.1 to generate samples from the perturbed pmf. In this case, the samples will be distributed according to a suboptimal pmf. The problem of optimal pmfs p^{opt} resulting in reducible Markov chains is a topic for future research.

3.4 Numerical examples

In this section, we will introduce numerical examples to illustrate the proposed input design method.

Example 3.1 *(Input design for nonlinear FIR models) In this example we will solve the input design problem for the system in Figure 3.1 in page 38, with*

$$G_0(u_t) = G_1(q,\theta)\, u_t + G_2(q,\theta)\, u_t^2 \, , \tag{3.22}$$

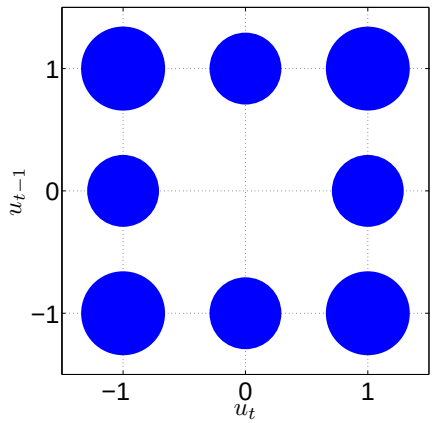

Figure 3.2: Plot with the stationary probabilities for the optimal input signal of Example 3.1. The radius of each disc is proportional to the probability of the state $(u_t,\, u_{t-1})$.

where

$$G_1(q,\theta) = \theta_1 + \theta_2\, q^{-1}\,, \tag{3.23}$$

$$G_2(q,\theta) = \theta_3 + \theta_4\, q^{-1}\,. \tag{3.24}$$

We assume that $\{e_t\}$ *is Gaussian white noise with variance* $\lambda_e = 1$. *This system has been introduced as an example in [54].*

We will solve Problem 3.2 by considering $h(\cdot) = \log\det(\cdot)$, *and a ternary sequence (*$n_{\mathcal{C}} = 3$*) of length* $n_m = 2$. *For this example, we take* $\mathcal{C} = \{-1,\, 0,\, 1\}$.

To solve (3.19)-(3.20) *we consider* $N_{\text{sim}} = 5 \cdot 10^3$ *in* (3.17). *The implementation of* (3.19)-(3.20) *was made in* `Matlab` *using the* `cvx` *toolbox [35].*

The simulation results give an optimal cost[2] $\log\det(\mathcal{I}_F^{\text{app}}) = -1.717$. *Figure 3.2 shows the optimal stationary probabilities for each state* $(u_t,\, u_{t-1})$ *(cf. Figure 4(a) in [54]). The results presented here show that the proposed method is consistent with the results in [54].*

Example 3.1 shows that this method is equivalent to the method introduced in [54] when G_0 has a nonlinear FIR-type structure.

The results in this chapter can also be employed for linear systems when amplitude constraints are considered in the input sequence by forcing u_t to belong to a finite alphabet. The next example shows an application in that direction.

[2] cf. $\log\det(P^{-1}) = -1.715$ for the same example in [54].

Example 3.2 *(Input design with amplitude constraints) In this example we consider the mass-spring-damper system introduced in [7]. The input, u, is the force applied to the mass and the output, y, is the mass position. The continuous-time system is described by the transfer function*

$$G_0(s) = \frac{\frac{1}{m}}{s^2 + \frac{c}{m}\, s + \frac{k}{m}}\,, \tag{3.25}$$

with $m = 100$ [Kg], $k = 10$ [N/m], and $c = 6.3246$ [Ns/m]. This choice results in the natural frequency $\omega_n = 0.3162$ [rad/s] and damping $\xi = 0.1$. The noise $\{e_t\}$ is white with zero mean and variance $\lambda_e = 10^{-4}$. The system (3.25) is sampled by using a zero-order-hold with sampling period $T_s = 1$ [s]. This gives the discrete-time system

$$G_0(u_t) = \frac{4.86 \cdot 10^{-3}\, q^{-1} + 4.75 \cdot 10^{-3} q^{-2}}{1 - 1.84\, q^{-1} + 0.94 q^{-2}}\, u_t\,. \tag{3.26}$$

As a model, we define

$$G(u_t;\, \theta) = \frac{\theta_1\, q^{-1} + \theta_2 q^{-2}}{1 + \theta_3\, q^{-1} + \theta_4 q^{-2}}\, u_t\,, \tag{3.27}$$

where

$$\theta = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 \end{bmatrix}^\top\,. \tag{3.28}$$

We will solve Problem 3.2 for two cost functions: $h(\cdot) = -\operatorname{tr}\{(\cdot)^{-1}\}$ and $h(\cdot) = \log\det(\cdot)$, subject to a binary sequence ($n_{\mathcal{C}} = 2$) of length $n_m = 2$. In this example, we define $\mathcal{C} = \{-1,\, 1\}$, and $N_{\text{sim}} = 5 \cdot 10^3$.

The solution of Problem 3.2 for this example gives $\operatorname{tr}\{(\mathcal{I}_F^{\text{app}})^{-1}\} = 0.1108$ and $\log\det(\mathcal{I}_F^{\text{app}}) = 28.22$. Figure 3.3 and 3.4 present the stationary probabilities of the optimal input signal for both cost functions. We can see that the stationary probabilities depend on the cost function h. However, both cost functions assign higher stationary probabilities to the states $(-1,\, -1)$ and $(1,\, 1)$.

We can compare the performance of our approach with the method introduced in [7]. For this purpose, we generate an input sequence of length N_{sim} by running the Markov chain associated to the stationary distribution in Figure 3.3, and the 4-states Markov chain presented in [7]. To guarantee that the input is a realization of a stationary process, we discard the first 10^6 realizations of the Markov chain. The results for the sampled information matrix are $\operatorname{tr}\{\mathcal{I}_F^{-1}\} = 1.8233 \cdot 10^{-4}$ for the 4-states Markov chain presented in [7], and $\operatorname{tr}\{\mathcal{I}_F^{-1}\} = 1.6525 \cdot 10^{-4}$ for our method (we note that our results are consistent with those reported in [7], since the scaling factor N_{sim} is not considered here). Therefore, based on the variance of the parameter estimates, we conclude that the approach in this chapter gives better results for the example introduced in [7].

To have an idea of the computation time required for this example, the optimization was solved in a laptop Dell Latitude E6430, equipped with Intel Core i7

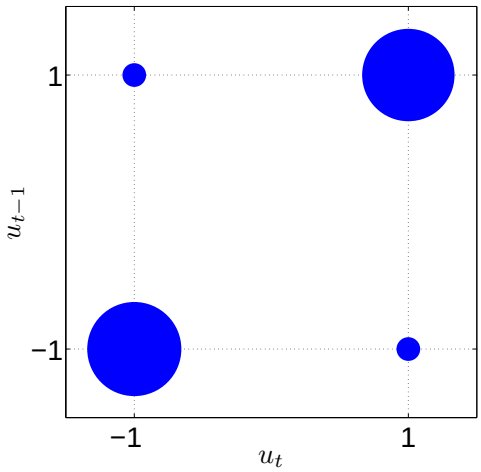

Figure 3.3: Plot with the stationary probabilities for the optimal input signal in Example 3.2 as in Figure 3.2. $h(\cdot) = -\operatorname{tr}\{(\cdot)^{-1}\}$.

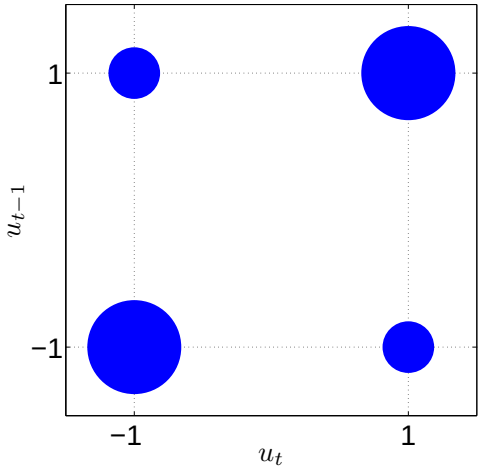

Figure 3.4: Plot with the stationary probabilities for the optimal input signal in Example 3.2 as in Figure 3.2. $h(\cdot) = \log\det(\cdot)$.

2.6 *[GHz] processor, and* 8 *[Gb] of RAM memory. The time required for the computation of elementary cycles to the computation of stationary probabilities is* 1.9 *seconds.*

The results presented in the previous examples show that the method introduced in this chapter retrieves (or improves) the results in the literature. The next examples show an application of the input design method to a model structure not covered by existing techniques.

Example 3.3 *(Input design for nonlinear output-error models) Consider the block diagram depicted in Figure 3.1, where* $\{e_t\}$ *is Gaussian white noise sequence with zero mean and variance* $\lambda_e = 1$*. The system is described by*

$$G_0(u_t) = \begin{cases} x_{t+1} = \dfrac{1}{\theta_1^0 + x_t^2} + u_t \,, \\ \\ z_t = \theta_2^0 \, x_t^2 \,, \\ \\ x_1 = 0 \,, \end{cases} \tag{3.29}$$

where $\theta_0 = \begin{bmatrix} \theta_1^0 & \theta_2^0 \end{bmatrix}^\top = \begin{bmatrix} 0.8 & 2 \end{bmatrix}^\top$*. Notice that the system* (3.29) *cannot be described as a Wiener-Hammerstein system, since the state equation is nonlinear in the state* x_t*.*

To apply our method, we consider the model

$$G(u_t; \theta) = \begin{cases} x_{t+1} = \dfrac{1}{\theta_1 + x_t^2} + u_t \,, \\ \\ z_t = \theta_2 \, x_t^2 \,, \\ \\ x_1 = 0 \,, \end{cases} \tag{3.30}$$

with $\theta = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^\top = \theta_0$*.*

We design an input of $n_{\text{seq}} = 10^4$ *samples as a realization of the pmf obtained by solving Problem 3.2, with* $n_m = 1$*, and* $\mathcal{C} = \{-1, 0, 1\}$*. Problem 3.2 will be solved for* $h(\cdot) = \log\det(\cdot)$*.*

The stationary probabilities $\mathbf{P}\{l\}$ *obtained as the solution of Problem 3.2 are presented in Figure 3.5. In this figure, we notice that the stationary probabilities cannot be associated with samples obtained from a uniform distribution among the states.*

Given the stationary probabilities in Figure 3.5, we can use Algorithm 2.1 to design a transition matrix to generate samples with the desired distribution. The resulting transition matrix is given by

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0.47 & 0.47 & 0.47 \\ 0.53 & 0.53 & 0.53 \end{bmatrix} , \tag{3.31}$$

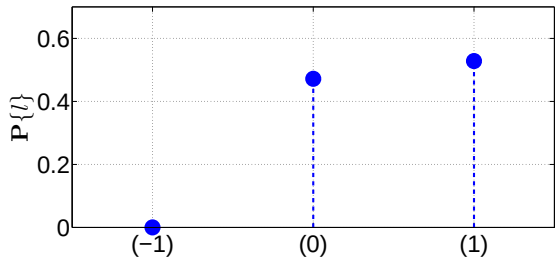

Figure 3.5: Plot with the stationary probabilities for the optimal input signal in Example 3.3.

Table 3.1: Numerical results for the cost function in Example 3.3.

$h(\mathcal{I}_F)$	Graph	Uniform	Normal	Binary
$\log\{\det(\mathcal{I}_F)\}$	3.67	2.77	3.12	3.47

where the rows and columns of A are indexed in the order (-1), (0), (1).

The second largest eigenvalue modulus (SLEM) for A defined in (3.31) *is* 0, *and the number of nonzero eigenvalues is* 1, *as expected by Theorem 2.3 in page 34.*

Finally, we generate the input signal by running the Markov chain with transition matrix (3.31) *with random initial state in* $\{0, 1\}$, *and recording the first* $n_{\text{seq}} = 10^4$ *samples. This is possible since the SLEM associated with the transition probability matrix* (3.31) *is zero, which implies that the chain will start in the stationary distribution. To compare the result of the new method with those of standard input signals, in Table 3.1 we present the results obtained for the cost function when the input is designed with the proposed method (*$\log\{\det(\mathcal{I}_F)\} = 3.67$ *in Graph), and when the input sequence is a realization of a sequence of independent and uniformly distributed random variables with support* $[-1, 1]$ *(*$\log\{\det(\mathcal{I}_F)\} = 2.77$ *in Uniform). The results show that the proposed method for input design outperforms the experiment result based on uniform samples. In addition, we can also compare our result with those obtained with the input being Gaussian distributed white noise, with zero mean and variance* 1 *(*$\log\{\det(\mathcal{I}_F)\} = 3.12$ *for Normal in Table 3.1), and when the input is a realization of a binary white noise process with values* -1 *and* 1 *(*$\log\{\det(\mathcal{I}_F)\} = 3.47$ *for Binary in Table 3.1). In this case, the results obtained by the Gaussian and the Binary distributed inputs are closer to the ones obtained by the proposed input design method. However, our method still performs better than the input sequence based on random samples.*

As an additional exercise, we can also compute the results obtained when the input is designed for the following cases:

$$\textit{Case 1: } n_m = 2,\ \mathcal{C} = \{-1,\ 0,\ 1\}\,, \tag{3.32}$$

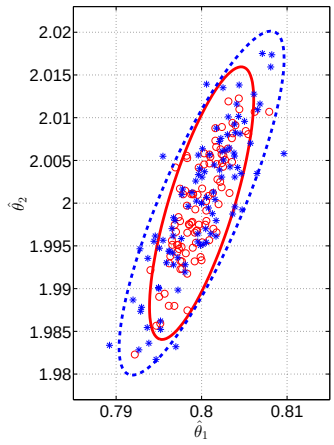

Figure 3.6: Plot with the 95 % confidence ellipsoids for input sequences of length $n_{\text{seq}} = 10^4$, and the estimated parameters, Example 3.3. Blue, dashed line: Confidence ellipsoid for a binary input sequence (realizations marked with $*$). Red, continuous line: Confidence ellipsoid for the optimal input sequence, Case 2 (realizations marked with circles).

Table 3.2: Numerical results for the cost function in Example 3.3, Cases 1-3.

$h(\mathcal{I}_F)$	Case 1	Case 2	Case 3
$\log\{\det(\mathcal{I}_F)\}$	3.82	4.50	4.48

$$\textit{Case 2: } n_m = 1,\ \mathcal{C} = \{-1,\ -1/3,\ 1/3,\ 1\}\,, \tag{3.33}$$
$$\textit{Case 3: } n_m = 1,\ \mathcal{C} = \{-1,\ -0.5,\ 0,\ 0.5,\ 1\}\,. \tag{3.34}$$

Table 3.2 presents the results when the Markov chain associated with the Cases 1-3 is employed to generate $n_{\text{seq}} = 10^4$ samples. From these results we conclude that we can increase the information obtained from the input if we extend the memory of the stationary process generating the input sequence. Moreover, the results are significantly better when we only extend the set $\mathcal{C}$, which can be seen by comparing the value of h of Case 1 with Cases 2 and 3 in Table 3.2. On the other hand, we observe an interesting result in Table 3.2. In contrast to what we might expect, an increase of the magnitude of the elements in $\mathcal{C}$ does not necessarily imply an increase of $\log\{\det(\mathcal{I}_F)\}$. This phenomenon can be explained by the structure of the system (3.29), where we see that an increase in the amplitude of u_t does not necessarily imply that we obtain more information from the system. Since x_{t+1} depends on x_t^{-2}, then the future state will be small if the present state has a large

value. Indeed, by comparing the value of h obtained in Case 2 and Case 3 in Table 3.2, we see that increasing the magnitude of the possible values for the input does not imply an increase in the value of h.

To analyze the accuracy of the method, we present in Figure 3.6 the 95 % *confidence ellipsoids for the input sequence generated from a random binary distribution, and from an optimal input obtained by solving Case 2. In addition, we also plot* 10^2 *estimated parameters computed using the data set generated with both input sequences. The results in the figure show that the proposed input design technique decreases the uncertainty region for the estimated parameters, compared to a random input sequence of length* n_{seq}*. This conclusion is also confirmed by the numerical estimates of* θ_0*, which obey the distribution given by the theoretical bounds. Therefore, the proposed technique is an attractive alternative to increase the accuracy of the parameter estimates over the one obtained with random samples.*

Example 3.4 *(Input design for nonlinear output-error models revisited) Consider the problem introduced in Example 3.3. As before, we will design an input of* $n_{\text{seq}} = 10^4$ *samples as a realization of the pmf obtained by solving Problem 3.2, for Case 2 and Case 3 in Example 3.3. Problem 3.2 is solved for* $h(\cdot) = -\operatorname{tr}\{(\cdot)^{-1}\}$.

Table 3.3: Numerical results for the cost function in Example 3.4.

$h(\mathcal{I}_F^s)$	Case 2	Case 3	Unif.	Normal	Binary
$-\operatorname{tr}\{\mathcal{I}_F^{s\,-1}\}$	-0.49	-0.45	-1.21	-0.96	-0.83

The results obtained by different input sequences of length $n_{\text{seq}} = 10^4$ *are presented in Table 3.3, where Unif., Normal and Binary represent the results obtained with the random samples defined in Example 3.3. The results show that the proposed input design technique (Case 2 and Case 3 in Table 3.3) outperforms the inputs based on white noise samples.*

To analyze the accuracy in this example, Figure 3.7 presents the 95 % *confidence ellipsoids for the input sequence generated from a random binary distribution, and from an optimal input obtained by solving Case 3. In addition, we also plot* 10^2 *estimated parameters computed by using the data set generated with both input sequences. In the same line as Figure 3.6, we see that the accuracy of the estimated parameters is improved for the confidence sets, when we compare the optimal input sequence with a white noise realization from a binary distribution. Therefore, the method presented in this chapter is an effective approach to design input sequences to identify the system* (3.29).

Remark 3.5 *The input design method presented in this chapter is close to the techniques introduced in [18, 26]. In those approaches, the input is restricted to the set* $\mathcal{P}_{\mathcal{C}}$ *introduced in Chapter 2. However, the results in [18, 26] do not exploit the*

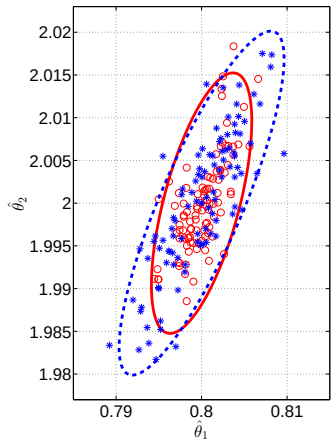

Figure 3.7: Plot with the 95 % confidence ellipsoids for input sequences of length $n_{\text{seq}} = 10^4$, and the estimated parameters, Example 3.4. Blue, dashed line: Confidence ellipsoid for a binary input sequence (realizations marked with $*$). Red, continuous line: Confidence ellipsoid for the optimal input sequence, Case 2 (realizations marked with circles).

connection between de Bruijn graphs and stationary processes, as it is presented in this thesis, which allows to generate an input realization from an element in $\mathcal{P}_C$.

3.5 Conclusion

In this chapter, a new input design technique for identification of nonlinear output-error models has been described. The method considers the design of an input sequence as a realization of a stationary process, which is obtained by maximizing a scalar cost function of the information matrix.

To obtain a computationally tractable problem, we parameterize the set of stationary process in terms of convex combinations of the extreme points of the set. By assuming that the stationary process has finite memory and finite alphabet, the probability measures associated with the extreme points are computed as the set of prime cycles associated with the equivalent de Bruijn graph. Therefore, the information matrix associated with any element in the set of stationary processes can be computed as a convex combination of the information matrices obtained for each extreme point. Due to the complexity of nonlinear model structures, the information matrices for each extreme point are obtained by numerical approximations. The main advantage of this approach is that the input design problem becomes convex even for nonlinear model structures.

Finally, this chapter introduced numerical examples to illustrate the effectiveness of the proposed input design approach. The numerical examples show that the input design method improves existing results in the literature. In addition, the proposed method can be employed to design input sequences to identify nonlinear model structures that are not covered by previous approaches.

Chapter 4

Optimal input design for nonlinear state space models

In Chapter 3 a new method for input design for identification of nonlinear models was introduced. In that chapter, the discussion was restricted to nonlinear output-error models. The assumption on the model structure was introduced to simplify the approximation of the Fisher information matrix, which is based on numerical approximations. However, the method in Chapter 3 does not cover the case where the noise process also affects the internal states of the model. In these models, the approximation of the Fisher information matrix given in Chapter 3 is no longer valid. Therefore, new approximation methods must be considered to extend the results in Chapter 3 to more general nonlinear model structures.

In this chapter we provide an extension of the input design method introduced in Chapter 3 to nonlinear state space models. As in Chapter 3, we use a graph theoretical approach to design an input sequence as a realization of a stationary process, which maximizes a scalar cost function of the Fisher information matrix. To estimate the Fisher information matrices for the extreme points in the set of stationary processes, we use particle methods to obtain the required estimations as the sample covariance matrix of the score function. Then the optimization is solved for different realizations of the sample covariance matrix of the score function, and the optimal pdf is obtained as the sample mean over the solutions obtained for the different data realizations. Numerical examples show that the method is an attractive approach to design input sequences for identification of nonlinear state space models.

4.1 Problem formulation

In this section[1] we introduce an extension of the optimal input design formulation presented in Chapter 3 to nonlinear state space models. The discussion in this part follows the same lines as Section 3.1, with the difference that here we formulate the problem to cover nonlinear state space models.

As in Chapter 3, the objective is to design an input signal $u_{1:n_{\text{seq}}} = (u_1, \ldots, u_{n_{\text{seq}}})$, as a realization of a stationary process. This is done so that a state space model (SSM) can be identified with maximum accuracy as defined by a scalar function of the Fisher information matrix $\mathcal{I}_F$ [59]. An SSM with states $x_{0:T} = (x_0, \ldots, x_T)$, inputs $u_{1:T}$ and measurements $y_{1:T}$ is given by

$$x_0 \sim \mu(x_0), \tag{4.1a}$$
$$x_t|x_{t-1} \sim f_\theta(x_t|x_{t-1}, u_{t-1}), \tag{4.1b}$$
$$y_t|x_t \sim g_\theta(y_t|x_t, u_t). \tag{4.1c}$$

Here, $f_\theta(\cdot)$ and $g_\theta(\cdot)$ denote known probability distributions parametrized by $\theta \in \Theta \subset \mathbb{R}^{n_\theta}$. In the sequel, we make the rather restrictive albeit standard assumption that there exists $\theta_0 \in \Theta$ such that the model (4.1) describes the system to be identified when $\theta = \theta_0$, i.e., there is no undermodeling. This assumption is necessary in order to quantify the information retrieved from the experiment as a function of the Fisher information matrix.

We notice that we can write the joint distribution of states and measurements for (4.1) as

$$p_\theta(x_{1:T}, y_{1:T}|u_{1:T}) = \mu(x_0) \prod_{t=1}^{T} f_\theta(x_t|x_{t-1}, u_{t-1}) g_\theta(y_t|x_t, u_t). \tag{4.2}$$

This quantity is used in the sequel for estimating $\mathcal{I}_F$ by

$$\mathcal{I}_F = \mathbf{E}\left\{\mathcal{S}(\theta_0)\mathcal{S}^\top(\theta_0)\right\}, \tag{4.3a}$$
$$\mathcal{S}(\theta_0) = \nabla_\theta \log p_\theta(y_{1:n_{\text{seq}}})\big|_{\theta=\theta_0}, \tag{4.3b}$$

where $p_\theta(y_{1:n_{\text{seq}}})$ and $\mathcal{S}(\theta)$ denote the likelihood function and the score function, respectively. Note that the expected value in (4.3a) is with respect to the probability distribution $p_\theta(x_{0:T}|y_{1:T})$ and the realizations of $u_{1:n_{\text{seq}}}$.

We note that (4.3a) depends on the cumulative distribution function of $u_{1:n_{\text{seq}}}$, $P_u(u_{1:n_{\text{seq}}})$. Therefore, the input design problem is to find a cdf $P_u^{\text{opt}}(u_{1:n_{\text{seq}}})$ which maximizes a scalar function of (4.3a). We define this scalar function as $h : \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}$. To obtain the desired results, h must be a concave, nondecreasing, matrix function [5, pp. 108] (cf. Section 3.1). In this chapter we leave the selection of h to the user.

[1] Throughout this chapter we introduce a subindex in the cdfs and pdfs to emphasize the random variables considered in the distribution.

Since $P_u^{\text{opt}}(u_{1:n_{\text{seq}}})$ is assumed to describe a stationary Markov process with memory n_{seq}, the optimization must be constrained to the set

$$\mathcal{P} = \{P_u : \mathbb{R}^{n_{\text{seq}}} \to \mathbb{R} |\, P_u(\mathbf{x}) \geq 0,\, \forall \mathbf{x} \in \mathbb{R}^{n_{\text{seq}}};\; P_u \text{ is monotone nondecreasing};\; \lim_{\substack{x_i \to \infty \\ i=\{1,\ldots,n_{\text{seq}}\} \\ \mathbf{x}=(x_1,\ldots,x_{n_{\text{seq}}})}} P_u(\mathbf{x}) = 1;\; \int_{v\in\mathbb{R}} dP_u([v,\, \mathbf{z}]) = \int_{v\in\mathbb{R}} dP_u([\mathbf{z},\, v]),\, \forall \mathbf{z} \in \mathbb{R}^{n_{\text{seq}}-1} \Big\}. \quad (4.4)$$

As in Chapter 3, here we restrict our problem to u_t belonging to an alphabet with finite cardinality $n_{\mathcal{C}}$. Thus, the set of pmfs associated with these processes is

$$\mathcal{P}_{\mathcal{C}} = \{p_u : \mathcal{C}^{n_{\text{seq}}} \to \mathbb{R} |\, p_u(\mathbf{x}) \geq 0,\, \forall \mathbf{x} \in \mathcal{C}^{n_{\text{seq}}};\; \sum_{\mathbf{x}\in\mathcal{C}^{n_{\text{seq}}}} p_u(\mathbf{x}) = 1;\; \sum_{v\in\mathcal{C}} p_u([v,\, \mathbf{z}]) = \sum_{v\in\mathcal{C}} p_u([\mathbf{z},\, v]),\, \forall \mathbf{z} \in \mathcal{C}^{(n_{\text{seq}}-1)} \Big\}. \quad (4.5)$$

Finally, the discussion in this section can be summarized as

Problem 4.1 *Design an optimal input signal $u_{1:n_{\text{seq}}} \in \mathcal{C}^{n_{\text{seq}}}$ as a realization from $p_u^{\text{opt}}(u_{1:n_{\text{seq}}})$, where*

$$p_u^{\text{opt}} := \arg\max_{p_u \in \mathcal{P}_{\mathcal{C}}} h(\mathcal{I}_F(p_u)), \quad (4.6)$$

with $h : \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}$ a concave, nondecreasing, matrix function, and $\mathcal{I}_F \in \mathbb{R}^{n_\theta \times n_\theta}$ defined as in (4.3).

Before introducing the input design method, we review some material on sequential Monte Carlo (SMC) methods.

4.2 A review on sequential Monte Carlo methods

SMC methods are a family of techniques that can be used, e.g., to estimate the filtering and smoothing distributions in SSMs. Here we describe the idea behind SMC methods, and we introduce the required material to understand the estimation algorithms employed in this chapter. The material in this section is based on [19, 22, 76].

SMC methods: main idea

The objective of SMC methods is to sample sequentially from target probability densities $\{p(x_{1:k})\}_{k\geq 1}$ of increasing dimension, where $x_k \in \mathcal{X}$, and each distribution $p(x_{1:k})$ is defined on the product space $\mathcal{X}^k$. Writing

$$p(x_{1:k}) = \frac{\gamma_k(x_{1:k})}{Z_k}\,, \tag{4.7}$$

it is only required that $\gamma_k \colon \mathcal{X}^k \to \mathbb{R}^+$ is known pointwise; the normalizing constant

$$Z_k = \int_{\mathcal{X}^k} \gamma_k(x_{1:k}) dx_{1:k}\,, \tag{4.8}$$

might be unknown if γ_k is known pointwise. SMC provides an approximation of $p_1(x_1)$ and an estimate of Z_1 at time 1, then an approximation of $p_2(x_{1:2})$, and an estimate of Z_2 at time 2, and so on.

Monte Carlo methods

Consider initially approximating a generic probability distribution $p(x_{1:T})$ for some fixed T. If we sample N independent random vectors $\{x_{1:T}^{(i)}\}_{i=1}^N$ from $p(x_{1:T})$, then the Monte Carlo method approximates functions of $p(x_{1:T})$ in the mean square sense by using

$$\hat{p}(x_{1:T}) = \frac{1}{N}\sum_{i=1}^{N} \delta\left(x_{1:T} - x_{1:T}^{(i)}\right)\,, \tag{4.9}$$

where $\delta(x)$ denoted the Dirac delta function located at $x = 0$. Based on this approximation, it is possible to approximate any marginal $p(x_k)$ by

$$\hat{p}(x_k) = \frac{1}{N}\sum_{i=1}^{N} \delta\left(x_k - x_k^{(i)}\right)\,, \tag{4.10}$$

and the expectation of any measurable function $\varphi_T : \mathcal{X}^T \to \mathbb{R}$, given by

$$\mathbf{E}\left\{\varphi_T(x_{1:T})\right\} = \int_{\mathcal{X}^T} \varphi_T(x_{1:T}) p(x_{1:T})\, dx_{1:T}\,, \tag{4.11}$$

is estimated as

$$\hat{\mathbf{E}}^{\mathrm{MC}}\left\{\varphi_T(x_{1:T})\right\} := \int_{\mathcal{X}^T} \varphi_T(x_{1:T}) \hat{p}(x_{1:T})\, dx_{1:T} = \frac{1}{N}\sum_{i=1}^{N} \varphi_T\left(x_{1:T}^{(i)}\right)\,. \tag{4.12}$$

It is easy to check that (4.12) is an unbiased estimate of (4.11), and that its variance is given by

$$\mathrm{Cov}\left\{\hat{\mathbf{E}}^{\mathrm{MC}}\left\{\varphi_T(x_{1:T})\right\}\right\} = \frac{1}{N}\left\{\int_{\mathcal{X}^T} \varphi_T^2(x_{1:T}) p(x_{1:T})\, dx_{1:T} - \mathbf{E}\left\{\varphi_T(x_{1:T})\right\}\right\}\,. \tag{4.13}$$

The main advantage of Monte Carlo methods over standard approximation techniques is that the variance of the approximation error decreases at a rate of $\mathcal{O}(1/N)$ regardless of the dimension of the space $\mathcal{X}^T$. However, there are at least two main problems with this idea:

(i) If $p(x_{1:T})$ is a complex high-dimensional probability distribution, then we cannot sample directly from it due to computational limitations.

(ii) Even if we knew how to sample exactly from $p(x_{1:T})$, the computational complexity of such a sampling scheme is typically at least linear in the number of variables T. Therefore, an algorithm sampling exactly from $p(x_{1:T})$, sequentially for each value of T, would have a computational complexity increasing at least linearly with T.

Importance sampling

In this part we address Problem (i) using the importance sampling (IS) method. IS relies on the introduction of an *importance density* $p_{\mathrm{I}}(x_{1:T})$ such that

$$p(x_{1:T}) > 0 \Rightarrow p_{\mathrm{I}}(x_{1:T}) > 0\,,$$

i.e., the distributions satisfy $\mathrm{supp}(p_{\mathrm{I}}) \subseteq \mathrm{supp}(p)$. In this case, we have from (4.7)-(4.8) the following identities:

$$p(x_{1:T}) = \frac{w_T(x_{1:T})p_{\mathrm{I}}(x_{1:T})}{Z_T}\,, \tag{4.14}$$

$$Z_T = \int_{\mathcal{X}^T} w_T(x_{1:T})p_{\mathrm{I}}(x_{1:T})\,dx_{1:T}\,, \tag{4.15}$$

where $w_T(x_{1:T})$ is the *unnormalized weight* function

$$w_T(x_{1:T}) = \frac{\gamma_T(x_{1:T})}{p_{\mathrm{I}}(x_{1:T})}\,. \tag{4.16}$$

In particular, we can select an importance density $p_{\mathrm{I}}(x_{1:T})$ from which it is easy to draw samples, e.g., the normal distribution, if this is consistent with the support of p.

If we assume we draw N independent samples $\{x_{1:T}^{(i)}\}_{i=1}^N$ from $p_{\mathrm{I}}(x_{1:T})$, then by inserting the Monte Carlo approximation of $p_{\mathrm{I}}(x_{1:T})$ (the empirical distribution of the samples $\{x_{1:T}^{(i)}\}_{i=1}^N$) into (4.14)-(4.15), we have

$$\hat{p}(x_{1:T}) = \sum_{i=1}^{N} w_T^{(i)}\,\delta\left(x_{1:T} - x_{1:T}^{(i)}\right)\,, \tag{4.17}$$

$$\hat{Z}_T = \frac{1}{N}\sum_{i=1}^{N} w_T\left(x_{1:T}^{(i)}\right)\,, \tag{4.18}$$

where

$$w_T^{(i)} = \frac{w_T\left(x_{1:T}^{(i)}\right)}{\sum_{j=1}^{N} w_T\left(x_{1:T}^{(j)}\right)}, \tag{4.19}$$

denotes the normalized weights.

Based on the estimate (4.17), we can compute an estimate of (4.11) as

$$\hat{\mathbf{E}}^{\mathrm{IS}}\left\{\varphi_T(x_{1:T})\right\} := \int_{\mathcal{X}^T} \varphi_T(x_{1:T})\hat{p}(x_{1:T})\, dx_{1:T} = \sum_{i=1}^{N} w_T^{(i)} \varphi_T\left(x_{1:T}^{(i)}\right). \tag{4.20}$$

Unlike the estimate $\hat{\mathbf{E}}^{\mathrm{MC}}\left\{\varphi_T(x_{1:T})\right\}$, the expression (4.20) is in general biased for finite N. However, (4.20) is consistent in the number of samples N. The expression for the asymptotic bias is given by

$$\lim_{N\to\infty} N\left(\hat{\mathbf{E}}^{\mathrm{IS}}\left\{\varphi_T(x_{1:T})\right\} - \mathbf{E}\left\{\varphi_T(x_{1:T})\right\}\right) = - \int_{\mathcal{X}^T} \frac{p^2(x_{1:T})}{p_{\mathrm{I}}(x_{1:T})} \left(\varphi_T(x_{1:T}) - \mathbf{E}\left\{\varphi_T(x_{1:T})\right\}\right) d\, x_{1:T}\,. \tag{4.21}$$

Furthermore, $\hat{\mathbf{E}}^{\mathrm{IS}}\left\{\varphi_T(x_{1:T})\right\}$ satisfies

$$\sqrt{N}\left(\hat{\mathbf{E}}^{\mathrm{IS}}\left\{\varphi_T(x_{1:T})\right\} - \mathbf{E}\left\{\varphi_T(x_{1:T})\right\}\right) \in \mathrm{As}\mathcal{N}\left(0,\, \mathcal{M}\right), \tag{4.22}$$

where

$$\mathcal{M} := \int_{\mathcal{X}^T} \frac{p^2(x_{1:T})}{p_{\mathrm{I}}(x_{1:T})} \left(\varphi_T(x_{1:T}) - \mathbf{E}\left\{\varphi_T(x_{1:T})\right\}\right)^2 d\, x_{1:T}\,. \tag{4.23}$$

As we can see from (4.21)-(4.23), both the bias and the variance for this method are of order $\mathcal{O}(1/N)$.

The next sampling method addresses the issue related with the increasing computational complexity in the sample length T.

Sequential importance sampling

Sequential importance sampling (SIS) is a method that admits a fixed computational complexity at each time step, and thus addresses Problem (ii). This method considers an importance distribution which has the following structure:

$$\begin{aligned} p_{\mathrm{I}}(x_{1:T}) &= p_{\mathrm{I}}(x_{1:T-1}) p_{\mathrm{I}}(x_T|x_{1:T-1}) \\ &= p_{\mathrm{I}}(x_1) \prod_{t=2}^{T} p_{\mathrm{I}}(x_t|x_{1:t-1})\,. \end{aligned} \tag{4.24}$$

From a practical perspective, Equation (4.24) means that to obtain particles $x_{1:T}^{(i)}$ distributed according to $p_{\mathrm{I}}(x_{1:T})$ at time T, we sample $x_1^{(i)}$ from $p_{\mathrm{I}}(x_1)$ at time 1,

then $x_t^{(i)}$ is sampled from $p_{\mathrm{I}}(x_t|x_{1:t-1}^i)$ at time t, for $t \in \{2, \ldots, T\}$. The associated unnormalized weights can be computed recursively using the decomposition

$$\begin{aligned} w_T(x_{1:T}) &= \frac{\gamma_T(x_{1:T})}{p_{\mathrm{I}}(x_{1:T})} \\ &= \frac{\gamma_{T-1}(x_{1:T-1})}{p_{\mathrm{I}}(x_{1:T-1})} \cdot \frac{\gamma_T(x_{1:T})}{\gamma_{T-1}(x_{1:T-1})p_{\mathrm{I}}(x_T|x_{1:T-1})}\,, \end{aligned} \tag{4.25}$$

which can be written in the form

$$\begin{aligned} w_T(x_{1:T}) &= w_{T-1}(x_{1:T-1}) \cdot \alpha(x_{1:T}) \\ &= w_1(x_1) \prod_{k=2}^{T} \alpha(x_{1:k})\,, \end{aligned} \tag{4.26}$$

where the *incremental importance weight* function $\alpha(x_{1:T})$ is given by

$$\alpha(x_{1:T}) := \frac{\gamma_T(x_{1:T})}{\gamma_{T-1}(x_{1:T-1})p_{\mathrm{I}}(x_T|x_{1:T-1})}\,. \tag{4.27}$$

The SIS method is summarized in Algorithm 4.1.

Algorithm 4.1 provides the estimates $\hat{p}(x_{1:t})$ and $\hat{Z}_t$ (equations (4.17)-(4.18)) of $p(x_{1:t})$ and Z_t at any time t. In this framework, it seems that the only degree of freedom the user has at time t is the choice of $p_{\mathrm{I}}(x_t|x_{1:t-1})$ (the number of samples N is also a degree of freedom, but it is fixed before executing the algorithm). A sensible strategy consists of selecting it so as to minimize the variance of $w_t(x_{1:t})$. This is achieved by selecting

$$p_{\mathrm{I}}^{\mathrm{opt}}(x_t|x_{1:t-1}) = p(x_t|x_{1:t-1})\,, \tag{4.31}$$

as in this case the variance of $w_t(x_{1:t})$ conditional upon $x_{1:t-1}$ is zero, and the associated incremental weight is given by

$$\alpha^{\mathrm{opt}}(x_{1:t}) = \frac{\gamma_t(x_{1:t-1})}{\gamma_{t-1}(x_{1:t-1})} = \frac{\int_{\mathcal{X}} \gamma_t(x_{1:t})\,dx_t}{\gamma_{t-1}(x_{1:t-1})}\,. \tag{4.32}$$

We note that it is not always possible to sample from $p(x_t|x_{1:t-1})$, and nor is it always possible to compute $\alpha^{\mathrm{opt}}(x_{1:t})$. In these cases, we need to employ an approximation of $p_{\mathrm{I}}^{\mathrm{opt}}(x_t|x_{1:t-1})$ for $p_{\mathrm{I}}(x_t|x_{1:t-1})$.

In those scenarios in which the time required to sample from $p_{\mathrm{I}}(x_t|x_{1:t-1})$ and to compute $\alpha(x_{1:t})$ is independent of t (which is the case if p_{I} is chosen carefully and one is concerned with a problem such as filtering), it appears that SIS provides a solution for Problem (ii). However, the SIS method suffers from severe drawbacks. Even for standard IS, the variance of the resulting estimates increases exponentially with t [51]. A method to overcome this difficulty is presented in the next section.

Algorithm 4.1 Sequential importance sampling

INPUTS: N (number of samples), $p_{\mathrm{I}}(x_{1:T})$ (importance distribution), and $\gamma_T(x_{1:T})$ (unnormalized pdf).
OUTPUT: $\{x_{1:T}^{(i)}\}_{i=1}^N$ (realizations), and $\{w^{(i)}\}_{i=1}^N$ (normalized weights).

1: Sample $x_1^{(i)}$ from $p_{\mathrm{I}}(x_1)$ for $i = 1$ to N.
2: Compute the weights $w_1(x_1^{(i)}) = p_{\mathrm{I}}(x_1)$, and

$$w_1^{(i)} = \frac{w_1(x_1^{(i)})}{\sum_{j=1}^N w_1(x_1^{(i)})}, \tag{4.28}$$

for $i = 1$ to N.
3: **for** $t = 2$ to T **do**
4: Sample $x_t^{(i)}$ from $p_{\mathrm{I}}(x_t|x_{1:t-1}^{(i)})$ for $i = 1$ to N.
5: Compute the weights

$$w_t(x_{1:t}^{(i)}) = w_{t-1}(x_{1:t-1}^{(i)}) \cdot \alpha(x_{1:t}^{(i)}), \tag{4.29}$$

where α given in (4.32), and

$$w_t^{(i)} = \frac{w_t(x_{1:t}^{(i)})}{\sum_{j=1}^N w_t(x_{1:t}^{(i)})}, \tag{4.30}$$

for $i = 1$ to N.
6: **end for**

Resampling

We have seen that IS (and therefore SIS) provides estimates whose variance increases with t. Resampling techniques are a key ingredient of SMC methods which (partially) solve this problem in some important scenarios.

Resampling is an intuitive idea with major practical and theoretical benefits. We consider first an IS approximation $\hat{p}(x_{1:T})$ of the target distribution $p(x_{1:T})$. This approximation is based on weighted samples from $p_{\mathrm{I}}(x_{1:T})$, and does not provide samples approximately distributed according to $p(x_{1:T})$. To obtain approximate samples from $p(x_{1:T})$, we can simply draw samples from its IS approximation $\hat{p}(x_{1:T})$, where $\hat{p}(x_{1:T})$ is defined in (4.17), with normalized weights $\{w_T^{(i)}\}_{i=1}^N$. Then we select $x_{1:T}^{(i)}$ with probability $w_T^{(i)}$. This operation is called *resampling* as it corresponds to sampling from an approximation $\hat{p}(x_{1:T})$ which was itself obtained by sampling. If we are interested in obtaining N samples from $\hat{p}(x_{1:T})$, then we can resample N times from $\hat{p}(x_{1:T})$: This is equivalent to associating a number of offsprings $N_T^{(i)}$ with each sample $x_{1:T}^{(i)}$ in such a way that $N_T^{(1:N)} := \left(N_T^{(1)}, \ldots, N_T^{(N)}\right)$ follow a multinomial distribution with parameter vector $(N, w_T^{(1:N)})$, and associating a weight of $1/N$ with each offspring. Thus, we approximate $\hat{p}(x_{1:T})$ by the

Algorithm 4.2 Systematic resampling

INPUTS: $\{x_{1:T}^{(i)}\}_{i=1}^{N}$ (realizations), and $\{w_T^{(i)}\}_{i=1}^{N}$ (normalized weights).
OUTPUT: $\{x_{1:T}^{(i)}\}_{i=1}^{N}$ (realizations), and $\{N_T^{(i)}/N\}_{i=1}^{N}$ (resampled weights).

1: Sample $U_{(1)}$ from a uniform distribution defined on $[0,\, 1/N]$.
2: Define $U_{(i)} := U_{(1)} + (i-1)/N$ for $i \in \{2, \ldots, N\}$.
3: **for** $i = 1$ to N **do**
4: Set $N_T^{(i)} = \left| \left\{ U_{(j)} : \sum_{k=1}^{i-1} w_T^{(k)} \leq U_{(j)} \leq \sum_{k=1}^{i} w_T^{(k)} \right\} \right|$, where $\sum_{k=1}^{0} \mu_k := 0$.
5: **end for**

resampled empirical measure

$$\overline{p}(x_{1:T}) = \sum_{i=1}^{N} \frac{N_T^{(i)}}{N} \, \delta \left(x_{1:T} - x_{1:T}^{(i)} \right) , \tag{4.33}$$

where $\mathbf{E}\left\{ N_T^{(i)} \,\middle|\, w_T^{(1:N)} \right\} = N w_T^{(i)}$. Hence $\overline{p}(x_{1:T})$ is an unbiased approximation of $\hat{p}(x_{1:T})$.

Many resampling schemes have been proposed in the literature [22]. One of the most popular algorithms is *systematic resampling*, which is described in Algorithm 4.2. It can be shown that systematic resampling gives an unbiased estimate of the distribution $p(x_{1:T})$.

We clarify that resampling retrieves an estimate of $\mathbf{E}\{\varphi_T(x_{1:T})\}$ with higher variance than the one obtained by using $\hat{p}(x_{1:T})$. However, by resampling we remove samples with low weights with a high probability, which is useful when working with sequential techniques. If samples with low weights are preserved, then the approximation $\hat{p}(x_{1:T})$ will lose accuracy as we increase the path length T. This problem is known as particle degeneracy, which is discussed in more detail in page 65.

Remark 4.1 *The notion of offspring is not only important when talking about resampling, but it is also important for some of the particle methods discussed in the next sections. For future reference, we introduce the ancestor index $a_t^{(i)}$. Given a set of particles at time t, $\{x_t^{(k)}\}_{k=1}^{N}$, we denote by $a_t^{(i)}$ the index of the particle at time $t-1$ from which $x_t^{(i)}$ was generated. Figure 4.1 illustrates this idea. In Figure 4.1, the directed edges between two nodes at different time instants express which particles at time $t-1$ generate which particles at time t.*

Particle filterig

The name *particle filter* (PF) is a SMC method referring to the techniques employed to obtain estimates of the sequence of target densities $\{p(x_{1:t}|y_{1:t})\}_{t=1}^{T}$ or its marginal $\{p(x_t|y_{1:t})\}_{t=1}^{T}$. Here we assume that the underlying process is described for all $t \geq 1$ by equation (4.1).

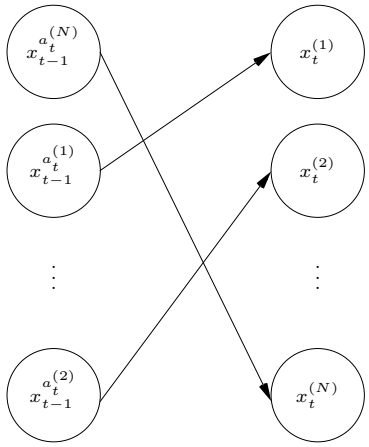

Figure 4.1: Illustration of the ancestor index $a_t^{(i)}$.

The basic idea is to leverage SIS and resampling to propagate a collection of N weighted random samples $\{x_t^{(i)}, w_t^{(i)}\}_{i=1}^N$ forward in time. These samples (commonly referred to as *particles*) constitute an empirical approximation that converges asymptotically to the underlying target density $p(x_t|y_{1:t})$ as $N \to \infty$.

The particle filter can be viewed as a framework to sequentially approximate the filtering densities $\{p(x_t|y_{1:t})\}_{t=1}^T$. The resulting approximation is an empirical distribution of the form

$$\hat{p}(x_t|y_{1:t}) = \sum_{i=1}^{N} w_t^{(i)}\, \delta(x_t - x_t^{(i)})\,. \tag{4.34}$$

Intuitively speaking, each particle $x_t^{(i)}$ can be understood as a possible state of the underlying system, where the corresponding weight $w_t^{(i)}$ contains information about how probable that particular state is.

In this subsection we introduce the method that will prove to be useful for the results presented in this chapter: the auxiliary particle filter (APF).

The auxiliary particle filter

The *auxiliary particle filter* (APF) [70] is a method which can be used to include the information available in the current observation y_t not only for proposing the new state x_t (as it is done in the PF method[2]), but also when proposing the ancestor index a_t (recall that a_t is the ancestor index of the particles at time t). Thus, we increase the probability of resampling particles at time $t-1$ that agree with the current observation y_t when compared with the PF approach.

[2]We refer to [22, 76] for more details of the particle filter algorithm.

To continue, we define a function $\nu : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, such that for each particle i, $i \in \{1, \ldots, N\}$ we compute the factors

$$\nu_{t-1}^{(i)} := \nu(x_{t-1}^{(i)}, y_t)\,, \tag{4.35}$$

referred to as *adjustment multipliers*. The adjustment multipliers are user defined functions, which are employed to adjust the empirical distributions to sample and propagate the particles. We note that (4.35) depends only on the previous particles and on the current observation, which are available in the resampling step at time t. The adjustment multipliers are then used to define a proposal distribution for the ancestor index a_t according to

$$\mathbf{P}\left\{a_t = i \,\middle|\, \left\{x_{t-1}^{(j)}, w_{t-1}^{(j)}\right\}_{j=1}^{N}\right\} = \frac{w_{t-1}^{(i)}\nu_{t-1}^{(i)}}{\sum_{l=1}^{N} w_{t-1}^{(l)}\nu_{t-1}^{(l)}}\,. \tag{4.36}$$

Once the ancestor indices are generated, we propagate the particles to time t by simulating $x_t^{(i)} \sim p_{\text{prop}}(x_t|\overline{x}_{t-1}^{(i)}, y_t)$ for $i \in \{1, \ldots, N\}$, where $p_{\text{prop}}(x_t|x_{t-1}, y_t)$ is the proposal distribution, and $\overline{x}_{t-1}^{(i)} = x_{t-1}^{(a_t)}$.

In addition to the elements introduced in this part, the APF needs the definition of a weight function from which $w_t^{(i)}$ are generated. For the APF, the weight function $w : \mathcal{X}^2 \times \mathcal{Y} \to \mathbb{R}$ is defined as[3]

$$w(x_{t-1}, x_t, y_t) = \frac{g(y_t|x_t)\, f(x_t|x_{t-1})}{\nu(x_{t-1}, y_t)\, p_{\text{prop}}(x_t|x_{t-1}, y_t)}\,. \tag{4.37}$$

Algorithm 4.3 summarizes the auxiliary particle filter method. We notice that, at time t, the APF performs three steps: (i) resampling of the particles at time $t-1$, (ii) propagation of the particles from time $t-1$ to time t, and (iii) normalized weighting of the set of particles at time t. These steps are a common framework of particle filter methods [76].

Particle degeneracy

In principle, the SMC can be used to obtain an approximation of a sequence of densities $\{p(x_{1:t}|y_{1:t})\}_{t\geq 1}$ of growing dimension, where the approximations tends to the true distributions asymptotically in the number of particles N, i.e., as $N \to \infty$. However, the task of approximating $\{\hat{p}(x_{1:t}|y_{1:t})\}_{t\geq 1}$ is inherently impossible to solve using a finite number of particles N. To illustrate this, assume that we employ a particle filter to target the joint smooth density $p(x_{1:t}|y_{1:t})$. At time s we generate N unique[4] particles $\{x_s^{(i)}\}_{i=1}^N$ from the proposal density, and we append these to the existing particle trajectories $\{x_{1:s-1}^{(i)}\}_{i=1}^N$. Therefore, we have a

[3]We note that the PF is obtained from the APF when $\nu(x_{t-1}, y_t) = 1$ [76].

[4]By unique we mean that each particle $x_s^{(i)}$ is a different realization from the proposal density, for $i \in \{1, \ldots, N\}$.

Algorithm 4.3 Auxiliary particle filter (APF)

1: **Initialization** ($t = 1$)**:**
2: Sample $x_1^{(i)} \sim p_{\text{prop}}(x_1|y_1)$.
3: Compute the importance weights $\tilde{w}_1^{(i)} = g(y_1|x_1^{(i)})\mu(x_1^{(i)})/p_{\text{prop}}(x_1^{(i)}|y_1)$, and normalize $w_1^{(i)} = \tilde{w}_1^{(i)} / \sum_{j=1}^N \tilde{w}_1^{(j)}$.
4: **for** $t = 2$ to T **do**
5: Compute the adjustment multipliers $\nu_{t-1}^{(i)} = \nu(x_{t-1}^{(i)}, y_t)$.
6: **Resampling:** Resample $\{x_{t-1}^{(i)}\}_{i=1}^N$ with probabilities proportional to $\{w_{t-1}^{(i)}\nu_{t-1}^{(i)}\}_{i=1}^N$ to generate the equally weighted particle system $\{\overline{x}_{t-1}^{(i)}, 1/N\}_{i=1}^N$.
7: **Propagation:** Sample $x_t^{(i)} \sim p_{\text{prop}}(x_t|\overline{x}_{t-1}^{(i)}, y_t)$.
8: **Weighting:** Compute $\tilde{w}_t^{(i)} = w(\overline{x}_{t-1}^{(i)}, x_t^{(i)}, y_t)$ and normalize:

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^N \tilde{w}_t^{(j)}}\,.$$

9: **end for**

weighted particle system $\{x_{1:s}^{(i)}, w_s^{(i)}\}_{i=1}^N$ approximating the joint smoothing density at time s. If we assume that the particle trajectories are resampled, then we obtain the particle system $\{\overline{x}_{1:s}^{(i)}, 1/N\}_{i=1}^N$. We recall that the purpose of resampling is to remove particles with small weights, and duplicate particles with large weights. Thus, the resampling step has the effect of reducing the number of unique particles. In consequence, over time each consecutive resampling of the particle trajectories will reduce the number of unique particles at time s. Eventually the particle system at time s will collapse into a single trajectory. This problem is referred to as *particle degeneracy*. In other words, the resampling step inevitable results in that for any time s there exists $t > s$ such that the PF approximation $\hat{p}(x_{1:t}|y_{1:t})$ consists of a single particle at time s.

A relevant question is to consider if it is possible (at least partly) to "undo" the particle degeneracy. One idea is to investigate if it is possible to initiate a backwards sweep starting from the particle filter representation of $p(x_t|y_{1:t})$, and reintroduce diversity among the particles by some form of backwards computation. This question[5] is addressed by a family of algorithms referred to as *particle smoothers*, which are introduced in the next subsection.

[5]Other solutions to the particle degeneracy problem involves increasing the number of particles N, or the fully-adapted PF [76].

Particle smoothing

As previously mentioned, if we only use the particle filter to compute the sequence of densities $\{p(x_{1:t}|y_{1:t})\}_{t\geq 1}$, then the approximations $\{\hat{p}(x_{1:t}|y_{1:t})\}_{t\geq 1}$ will have the particle degeneracy problem. To overcome this issue, we employ particle smoothers to compute the joint distribution. In this subsection, we introduce the fixed-lag smoother.

The fixed-lag smoother

The *fixed-lag* (FL) *smoother* [50] is an algorithm employed to estimate the smoothing densities, which decreases the particle degeneracy problem experienced with the particle filter. The fixed-lag smoother is used to approximate functionals in *additive form*. This means that, provided $\mathcal{F}\colon \mathcal{X}^T \to \mathbb{R}$ is a functional dependent on $x_{1:T}$, we can use the fixed-lag smoother to approximate expected values dependent on $\mathcal{F}$ if and only if $\mathcal{F}$ can be written as

$$\mathcal{F}(x_{1:T}) = \sum_{t=1}^{T-1} s_t(x_{t:t+1})\,, \tag{4.38}$$

where $\{s_t\}_{t\geq 1}$ is a sequence of measurable functions (which may depend on the observed data $y_{1:T}$) [68].

The key assumption in the fixed-lag smoother is that the pdf $p_\theta(x_t|y_{1:T})$ can be approximated by $p_\theta(x_t|y_{1:\kappa_t})$, for $\kappa_t = \min(t+\Delta, T)$ with some fixed-lag Δ, $0 \leq \Delta \leq T$. Therefore, the fixed-lag smoother assumes that the mixing properties of the process (4.1) are such that all the measurements $\{y_k\}_{k>\Delta}$ have a negligible effect on the conditional pdf of x_t given $y_{1:T}$, denoted by $p(x_t|y_{1:T})$.

The FL smoother employs the same algorithm as the APF (presented in Algorithm 4.3). However, the FL smoother approximates the two-step filtering distribution by

$$\hat{p}\left(x_{t-1:t}\right) = \sum_{t=1}^{T-1}\sum_{i=1}^{N} w_{\kappa_t}^{(i)}\,\delta\left(x_{t-1:t} - (x_{t-1}^{a_{\kappa_t,t}^{(i)}}, x_t^{a_{\kappa_t,t-1}^{(i)}})\right)\,, \tag{4.39}$$

where $a_{\kappa_t,t}^{(i)}$ denotes the particle at time t which is the ancestor of particle i at time κ_t.

It can be shown that, under some strict mixing assumptions, the variance and the bias of the FL smoother are of order $\mathcal{O}(T\log\{T\}/\sqrt{N})$ and $\mathcal{O}(T\log\{T\}/N)$ respectively. We refer to [68] for more details.

4.3 New input design method

In this section we will introduce a new input design method for nonlinear state space models.

Graph theoretical input design

Problem 4.1 will be solved using the graph theoretical approach presented in Chapter 3. To this end, we will consider $u_{1:n_{\text{seq}}}$ as a realization of a Markov process with memory n_m, where $n_m < n_{\text{seq}}$. Thus, we obtain a tractable description of the polytope $\mathcal{P}_{\mathcal{C}}$ in term of its extreme points. Following the notation introduced in Chapter 2, we will refer to $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}} = \{v_1, \ldots, v_{n_{\mathcal{V}}}\}$ as the set of the extreme points of $\mathcal{P}_{\mathcal{C}}$.

To find all the elements in $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$, we will make use of the results introduced in Section 2.2 in page 23. Indeed, we know that the set $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$ can be described in terms of the set of prime cycles in the equivalent de Bruijn graph $\mathcal{G}_{\mathcal{C}^{n_m}}$ (cf. Figure 2.1 in page 23). Moreover, the set of prime cycles in $\mathcal{G}_{\mathcal{C}^{n_m}}$ can be computed from the set of elementary cycles in $\mathcal{G}_{\mathcal{C}^{n_m-1}}$ (Lemma 2.2 in page 29). Finally, by using Theorem 2.1 in page 28, we have that the probability measures associated with the elements in $\mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$ are known once the set of prime cycles in $\mathcal{G}_{\mathcal{C}^{n_m}}$ is obtained.

Since we know the prime cycles in $\mathcal{G}_{\mathcal{C}^{n_m}}$, it is possible to generate an input sequence $\{u_t^j\}_{t=0}^T$ from v_j, which will be referred to as the *basis inputs*. We refer to Example 2.4 in page 30 for an illustration on how to obtain an input sequence from a given prime cycle.

Given $\{u_t^j\}_{t=0}^T$, we can use it to obtain the corresponding information matrix for $v_j \in \mathcal{V}_{\mathcal{P}_{\mathcal{C}}}$, say $\mathcal{I}_F^{(j)}$. However, in general the matrix $\mathcal{I}_F^{(j)}$ cannot be computed explicitly. To overcome this problem, we use SMC methods to approximate $\mathcal{I}_F^{(j)}$, as discussed in the next subsection.

Estimation of the score function

Consider (4.3). If samples of the score function were available we could approximate the Fisher information matrix (4.3) as the sample covariance matrix of the score function. However, the samples of the score function are not available. Therefore, we need to compute estimates associated with the score function. To this end, we will use SMC methods to estimate the score function, in particular we will use ideas from particle filter and fixed-lag particle smoothers.

As discussed in Section 4.2, the particle system is sequentially computed using two steps: (i) sampling/propagation and (ii) weighting. The first step can be seen as sampling from a proposal kernel,

$$\{a_t^{(i)}, x_t^{(i)}\} \sim \frac{\tilde{w}_{t-1}^{a_t}}{\sum_{k=1}^N \tilde{w}_{t-1}^{(k)}} R_{\theta,t}(x_t | x_{t-1}^{a_t}, u_{t-1}), \tag{4.40}$$

where we append the sampled particle to the trajectory by $x_{1:t}^{(i)} = \{x_{1:t-1}^{(i)}, x_t^{(i)}\}$. Here, $R_{\theta,t}(\cdot)$ denotes the propagation kernel[6] and the ancestor index $a_t^{(i)}$ denotes

[6]For the purpose of this subsection, the propagation kernel can be seen as the product $\nu(x_{t-1}, y_t)\, p_{\text{prop}}(x_t | x_{t-1}, y_t)$ in equation (4.37).

the index of the ancestor at time $t-1$ of particle $x_t^{(i)}$. In the second step, we calculate the (unnormalized) importance weights,

$$\tilde{w}_t^{(i)} \triangleq \frac{g_\theta(y_t|x_t^{(i)}, u_t) f_\theta(x_t|x_{t-1}^{(i)}, u_{t-1})}{R_{\theta,t}(x_t|x_{t-1}^{(i)}, u_{t-1})}. \tag{4.41}$$

We make use of Fisher's identity[7] [8, 25] to rewrite the score function (4.3b) into a form that can be used in combination with SMC methods. This results in that we can write

$$\nabla_\theta \log p_\theta(y_{1:T}|u_{1:T}) = \mathbf{E}\left\{\nabla_\theta \log p_\theta(x_{1:T}, y_{1:T}|u_{1:T}) \middle| y_{1:T}, u_{1:T}\right\}. \tag{4.42}$$

By using (4.2) into (4.42), and writing out the expected value, we obtain

$$\nabla_\theta \log p_\theta(y_{1:T}|u_{1:T}) = \sum_{t=1}^{T} \int_{\mathcal{X}^2} \xi_\theta(x_{t-1:t},\, u_t) p_\theta(x_{t-1:t}|y_{1:T})\, \mathrm{d}x_{t-1:t}, \tag{4.43}$$

with

$$\xi_\theta(x_{t-1:t},\, u_t) = \nabla_\theta \Big[\log f_\theta(x_t|x_{t-1}, u_{t-1}) + \log g_\theta(y_t|x_t, u_t) \Big],$$

which depends on the two-step marginal smoothing densities. We notice that the sum in (4.43) appears since the logarithm function transforms the product (4.2) into a sum. The APF can be used to estimate these quantities but this leads to poor estimates with high variance, due to problems with particle degeneracy.

Instead, we use an FL-smoother to estimate the smoothing densities, which reduces the variance of the score estimates [68]. We refer to [16] for more details about the FL-smoother and its use for score estimation. Based on the estimates for the target distributions obtained from Algorithm 4.3, the estimate of the score function is given by

$$\widehat{\mathcal{S}}(\theta) := \sum_{t=1}^{T} \sum_{i=1}^{N} w_{\kappa_t}^{(i)} \xi_\theta \Big(x_t^{a_{\kappa_t,t}^{(i)}}, x_{t-1}^{a_{\kappa_t,t-1}^{(i)}}, u_t \Big), \tag{4.44}$$

where $a_{\kappa_t,t}^{(i)}$ denotes the particle at time t which is the ancestor of particle i at time κ_t.

In the SMC literature, the Fisher information matrix is often estimated by the use of Louis' identity[8] [8, 62]. However, this approach does not guarantee that the information matrix estimate is positive semidefinite. Based on numerical experience, this standard approach also leads to poor accuracy.

Instead, we make use of the fact that the information matrix can be expressed as (4.3), i.e., the covariance matrix of the score function. Hence, a straightforward

[7] We refer the reader to Appendix C for a derivation of the Fisher's identity.
[8] We refer the reader to Appendix C for a derivation of Louis' identity.

method for estimating the information matrix is to use the Monte Carlo covariance estimator over some realizations of the system, given by $\{y_{1:T}^{(m)}\}_{m=1}^{M}$. If we denote each Monte Carlo estimate of the score function by $\widehat{\mathcal{S}}_m(\theta)$, the information matrix can be estimated using

$$\widehat{\mathcal{I}}_F = \frac{1}{M-1}\sum_{m=1}^{M} \widehat{\mathcal{S}}_m(\theta)\widehat{\mathcal{S}}_m^{\top}(\theta), \tag{4.45}$$

where M denotes the number of score estimates. Note that this is an estimate of the Fisher information matrix as the Monte Carlo estimator averages over the system realizations. The estimate is positive semidefinite by construction but inherits some bias from the FL-smoother, see [68] for more information. This problem can be handled by using a more computationally costly particle smoother. In Section 4.4 we present results indicating that this bias does not affect the resulting input signal to any large extent.

Solution to the optimization problem

As in Chapter 3, we associate $\{\mathcal{I}_F^{(j)}\}_{j=1}^{n_\mathcal{V}}$ with the elements in $\mathcal{V}_{\mathcal{P}_\mathcal{C}}$, the set of extreme points of $\mathcal{P}_\mathcal{C}$. By defining $\alpha := \{\alpha_1, \ldots, \alpha_{n_\mathcal{V}}\} \in \mathbb{R}^{n_\mathcal{V}}$, we introduce $\mathcal{I}_F^{\text{app}}(\alpha)$ as the information matrix associated with one element in $\mathcal{P}_\mathcal{C}$ for a given α such that $\alpha_j \geq 0$, $j \in \{1, \ldots, n_\mathcal{V}\}$, $\sum_{j=1}^{n_\mathcal{V}} \alpha_j = 1$. Therefore, finding the optimal $\mathcal{I}_F^{\text{app}}(\alpha)$ is equivalent to determining the optimal weighting vector α.

Hence, we can rewrite Problem 4.1 as

$$\alpha^{\text{opt}} = \arg\max_{\alpha \in \mathbb{R}^{n_\mathcal{V}}} h(\mathcal{I}_F^{\text{app}}(\alpha)), \tag{4.46a}$$

$$\text{s.t.} \quad \mathcal{I}_F^{\text{app}}(\alpha) = \sum_{j=1}^{n_\mathcal{V}} \alpha_j \mathcal{I}_F^{(j)}, \tag{4.46b}$$

$$\sum_{j=1}^{n_\mathcal{V}} \alpha_j = 1, \tag{4.46c}$$

$$\alpha_j \geq 0, \text{ for all } j \in \{1, \ldots, n_\mathcal{V}\}, \tag{4.46d}$$

To solve the optimization problem (4.46), we need to estimate the information matrix for each basis input.

The information matrix estimate in (4.45) can be used to estimate the Fisher information matrix for each basis input. A simple solution is therefore to plug-in the estimates and solve the convex optimization problem (4.46) using some standard solver. However, by doing this we neglect the stochastic nature of the estimates and disregard the uncertainty. In practice, this leads to bad estimates of α.

Instead, we propose the use of a Monte Carlo method which involves two different steps over K iterations. In step (a), we compute the information matrix

Algorithm 4.4 Optimal input design using SMC methods

INPUTS: Algorithm 4.3, K (no. MC runs) and M (size of each batch).
OUTPUT: $\alpha^\star$ (estimate of the optimal weighting of the basis inputs).

1: **for** $k = 1$ to K **do**
2: Generate M samples using Algorithm 4.3 for each basis input.
3: Estimate the information matrix by (4.45) for each basis input.
4: Solve the problem in (4.46).
5: Set $\alpha^{(k)}$ as the weighting factors obtained from the solver.
6: **end for**
7: Compute the sample mean of $\alpha = \{\alpha^{(1)}, \ldots, \alpha^{(K)}\}$, denote it as $\alpha^\star$.

estimates $\mathcal{I}_F^{(j)}$ for each input using (4.45). In step (b), we solve the optimization problem in (4.46) using the estimates $\{\widehat{\mathcal{I}}_F^{(j)}\}_{j=1}^{n_\mathcal{V}}$ to obtain the optimal weights $\alpha^{(k)}$ at iteration k. The estimate of the optimal weighting vector $\alpha^\star$ is computed as the sample mean of $\alpha = \{\alpha^{(1)}, \ldots, \alpha^{(K)}\}$, which can be complemented with a confidence interval. The outline of the complete procedure is presented in Algorithm 4.4.

Summary of the method

The proposed method for designing input signals in $\mathcal{C}^{n_m}$ considers the approach presented in Chapter 3, page 43, with the difference that the approximation of the Fisher information matrix in step 4 is performed using Algorithm 4.3, and the optimization is solved using Algorithm 4.4.

4.4 Numerical examples

The following examples present some applications of the proposed input design method.

Example 4.1 *Consider the linear Gaussian state space (LGSS) system with parameters* $\theta = \begin{bmatrix}\theta_1 & \theta_2\end{bmatrix}^\top$,

$$\begin{aligned} x_{t+1} &= \theta_1 x_t + u_t + v_t, & v_t &\sim \mathcal{N}(0, \theta_2^2), \\ y_t &= x_t + e_t, & e_t &\sim \mathcal{N}(0, 0.1^2), \end{aligned}$$

where the true parameters are $\theta_0 = \begin{bmatrix}0.5 & 0.1\end{bmatrix}^\top$. *We design experiments to identify* θ *with* $n_{\text{seq}} = 5 \cdot 10^3$ *time steps, memory length* $n_m = 2$, *and an input assuming values* $\mathcal{C} = \{-1, 0, 1\}$. *The optimal experiments maximize* $h(\mathcal{I}_F^{\text{app}}(\alpha)) = \log\det(\mathcal{I}_F^{\text{app}}(\alpha))$, *and* $h(\mathcal{I}_F^{\text{app}}(\alpha)) = -\operatorname{tr}\left\{(\mathcal{I}_F^{\text{app}}(\alpha))^{-1}\right\}$.

We generate $\{u_t^j\}_{t=0}^T$ *for each* $v_j \in \mathcal{V}_{\mathcal{P}_\mathcal{C}}$, $j \in \{1, \ldots, 8\}$ *(*$T = 10^2$*) to compute the approximation* (4.45) *for each* $\mathcal{I}_F^{(j)}$. *Finally, the optimal input* $u_{1:n_{\text{seq}}}$ *is computed by running a Markov chain with* $p_u^{\text{opt}}(u_{1:n_m})$ *as stationary pmf, where we discard*

Table 4.1: $h(\widehat{\mathcal{I}}_F)$, Example 4.1.

Input / $h(\widehat{\mathcal{I}}_F)$	$\log\det(\widehat{\mathcal{I}}_F)$	$-\operatorname{tr}\left\{(\widehat{\mathcal{I}}_F)^{-1}\right\}$
Optimal (log det)	20.67(0.01)	$-1.51 \cdot 10^{-4}(5.18 \cdot 10^{-7})$
Optimal (tr)	20.82(0.01)	$-1.32 \cdot 10^{-4}(4.45 \cdot 10^{-7})$
Binary	$\mathbf{20.91}(0.01)$	$\mathbf{-1.21 \cdot 10^{-4}}(4.51 \cdot 10^{-7})$
Uniform	19.38(0.01)	$-5.32 \cdot 10^{-4}(2.12 \cdot 10^{-6})$

the first $2 \cdot 10^6$ *samples and keep the last* $n_{\text{seq}} = 5 \cdot 10^3$ *ones. In addition, we consider* $K = 100$, $M = 5 \cdot 10^3$ *and* $N = 10^3$. *As a benchmark, we generate* n_{seq} *input samples from uniformly distributed white noise with support* $[-1, 1]$, *and the same amount of samples from binary white noise with values* $\{-1, 1\}$. *These input samples are employed to compute an approximation of* $\mathcal{I}_F$ *via* (4.45).

Table 4.1 condenses the results obtained for each input sequence, where Optimal (log det) *and* Optimal (tr) *represent the results for the input sequences obtained from optimizing* $\log\det(\mathcal{I}_F^{\text{app}}(\alpha))$, *and* $-\operatorname{tr}\left\{(\mathcal{I}_F^{\text{app}}(\alpha))^{-1}\right\}$, *respectively. The* 95% *confidence intervals are given as* $\pm$ *the value in the parentheses. From the data we conclude that, for this particular example, the binary white noise seems to be the best input sequence. Indeed, the proposed input design method tries to mimic the binary white noise excitation, which is clear from the numbers in Table 4.1. In this example, the results obtained with the proposed input design technique are degraded due to the variability in the optimization problem, which can be reduced by increasing the number of particles* N *in the estimation of the score function, by increasing the number of realizations* M *for the score function, and by increasing the number of iterations* K.

Example 4.2 *In this example we consider the system in [34, Section 6], given by*

$$\begin{aligned} x_{t+1} &= \theta_1 x_t + \frac{x_t}{\theta_2 + x_t^2} + u_t + v_t, & v_t &\sim \mathcal{N}(0, 0.1^2), \\ y_t &= \frac{1}{2}x_t + \frac{2}{5}x_t^2 + e_t, & e_t &\sim \mathcal{N}(0, 0.1^2), \end{aligned}$$

where $\theta = \begin{bmatrix}\theta_1 & \theta_2\end{bmatrix}^\top$ *denotes the parameters with true values* $\theta_0 = \begin{bmatrix}0.7 & 0.6\end{bmatrix}^\top$. *We design an experiment with the same settings as in Example 4.1 maximizing*

$$h(\mathcal{I}_F^{\text{app}}(\alpha)) = \log\det(\mathcal{I}_F^{\text{app}}(\alpha))\,.$$

A typical input realization obtained from the proposed input design method is presented in Figure 4.2.

Table 4.2 presents the results obtained for each input sequence, where Optimal *represents the result for the input sequence obtained from optimizing* $\log\det(\mathcal{I}_F^{\text{app}}(\alpha))$. *The* 95% *confidence intervals are given as* $\pm$ *the value in the parentheses. From these data we conclude that the extended input design method improves the experiment results obtained for binary and uniformly distributed samples.*

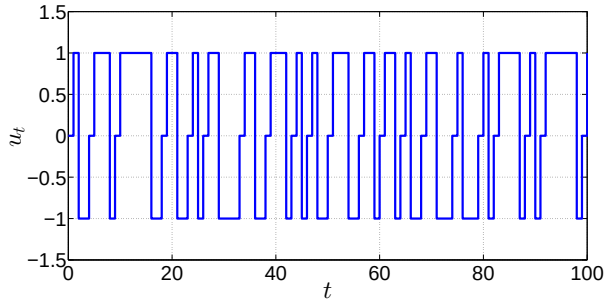

Figure 4.2: Input realization, Example 4.2.

Table 4.2: $h(\widehat{\mathcal{I}}_F)$, Example 4.2.

Input / $h(\widehat{\mathcal{I}}_F)$	$\log\det(\widehat{\mathcal{I}}_F)$
Optimal	**25.34**(0.01)
Binary	24.75(0.01)
Uniform	24.38(0.01)

4.5 Conclusion

In this chapter we discussed an extension of the input design method considered in Chapter 3. The extension considers a more general model structure in the form of a nonlinear state space description. Based on the approach introduced in Chapter 3, we design the input sequence by maximizing a scalar cost function of the Fisher information matrix over the set of stationary processes with finite memory and finite alphabet.

The main difficulty in this chapter is associated with the computation of the Fisher information matrix for the set of basis inputs. Since we assume a more general nonlinear model structure, the Monte Carlo approximation described in Chapter 3 cannot be employed here. To overcome this issue, we use particle methods to approximate the Fisher information matrices for the set of basis inputs. Since the method is based on numerical approximations, the resulting optimization is performed several times over different realizations, and the final result is the sample mean over the different realizations. Numerical examples show that the proposed technique can be employed for designing input sequences to identify nonlinear state space models.

As it might be expected, the computational complexity of this approach is not only related to the computation of the basis inputs, but it is also connected with the number of particles and the length of the input employed to estimate the score function. Future work in this subject will be related with the reduction of the computational complexity associated with the estimation of the score function.

Part II

Applications

Chapter 5

Input design for systems with quantized data

In this chapter, we present one application of the proposed input design method to identify systems subject to quantized measurements. This problem is of relevance because of the many applications where quantized measurements are used. For example, quantized measurements are widely used in networked control and wireless communications, due to the noise immunity of digital communication channels. In fact, due to constraints on the communication channels, such as noise and limited bandwidth, it is impossible to transmit data with infinite precision. Thus, quantization is an effective way to reduce use of the transmission resource, and meet the bandwidth constraint of a communication channel. However, quantization is a lossy compression method, hence the validity or performance of parameter identification may be affected by quantization. This could also result in the deteriorated performance of adaptive control techniques.

In wireless communications, for example, quantized system identification is crucial for echo cancellation and blind communication channel estimation, see [31] and the references therein. This means that quantized system identification is very important as the use of wireless technologies is now extensive in many areas of society, and it is expected to increase.

A particular type of quantized system is where the quantizer can take only 2 possible values (binary signals). This kind of configuration has attracted considerable attention [13, 48, 56]. In fact, binary-valued (or quantized) sensors are commonly used in practical systems. They are exemplified by switching sensors for exhaust gas oxygen, photoelectric sensors for position, Hall-effect sensors for speed and acceleration, gas content sensors in the gas and oil industry, and distributed one-bit wireless sensors [63, 71, 99]. In particular, for any remotely controlled system, signal quantization is mandatory. Compared to regular sensors, binary-valued sensors are usually far more cost effective. However, they provide very limited information on the process.

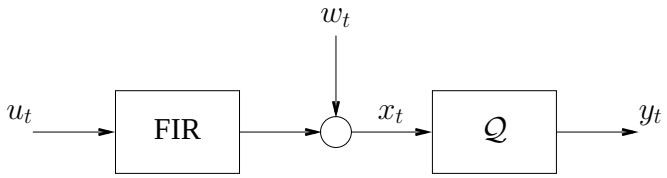

Figure 5.1: FIR system with quantized output.

The problem of input design for quantized systems has been treated in previous works [10, 11, 12, 98]. In [98] conditions under which consistent estimates can be obtained are presented. In particular, [98] recommends the use of periodic signals. Even though periodic inputs have shown to asymptotically achieve the Cramér-Rao lower bound (CRLB), their use may be inadequate for tracking control applications, since the output will not follow the desired trajectory [65].

In [12], the input design problem is carried out within a worst case framework, based on the set membership paradigm of uncertainty representation. There, the authors assume uncertainty in the parameters, and extend the previous work [11] from binary sensors to multilevel sensor thresholds.

The work [65] asserts that the input assuring strong consistency of the estimates is a persistently exciting signal of certain order r, as long as the quantizer is optimal in the sense of minimizing the error between the measurement and the input to the quantizer. However, the design of this signal is not carried out.

To use the input design technique proposed in previous chapters for identification of model structures with quantized output, we need an estimate of the Fisher information matrix (FIM) obtained when using the set of basis inputs. In this chapter an estimate of the FIM is given, based on the so-called Fisher's identity (cf. Subsection C.2 [66]). A numerical example shows that the proposed technique is suitable for solving the input design problem for identification of models with quantized outputs.

5.1 Input design problem

We focus on the problem of input design for quantized output systems. In particular, we have a system as in Figure 5.1, where noise enters between the linear FIR system and the quantizer. The noise, w_t, is assumed to be zero-mean Gaussian with variance σ^2. We also assume that σ^2 is known.

Remark 5.1 *Note that the assumption about the noise variance being known is only necessary for the experiment design, and not for the estimation problem. As shown in [31], the noise variance can also be estimated.*

The system can be described by

$$\begin{aligned} x_t &= \varphi_t^\top \theta_0 + w_t, \quad w_t \sim \mathcal{N}(0, \sigma^2) \\ y_t &= \mathcal{Q}[x_t], \end{aligned} \tag{5.1}$$

where x_t is a scalar, the parameter vector θ_0 is of dimension n_θ, φ_t is a vector of dimension n_θ, and $\sigma^2 \in \mathbb{R}$ is the noise variance. $\mathcal{Q} : \mathbb{R} \to \mathcal{V}$ represents a quantizer, which is a map from $\mathbb{R}$ the set of a scalars to the finite set $\mathcal{V} = \{v_1, v_2, \cdots, v_M\}$ of real numbers, i.e.,

$$\mathcal{Q}[x] = \begin{cases} v_1, & \text{if } x \in \Omega_1, \\ \vdots & \\ v_M, & \text{if } x \in \Omega_M, \end{cases} \tag{5.2}$$

where Ω_i, $i \in \{1, \ldots, M\}$ conform a partition of $\mathbb{R}$.

Since the system is described by (5.3), we consider the model set

$$\mathcal{M} = \left\{ \begin{aligned} x_t &= \varphi_t^\top \theta + w_t, \quad w_t \sim \mathcal{N}(0, \sigma^2) \\ y_t &= \mathcal{Q}[x_t], \end{aligned} \middle| \, \theta \in \Theta \right\}. \tag{5.3}$$

The problem of interest is to design the input $u_{1:n_{\text{seq}}} = (u_1,\ u_2,\ \ldots,\ u_{n_{\text{seq}}})$ such that the parameter θ_0 is better estimated compared to any other input signal. From (5.1) we see that the output y_t contains information about the parameter θ_0. Moreover, the information available in y_t depends on the vector φ_t, which depends on the input u_t. Therefore, the input sequence $u_{1:n_{\text{seq}}}$ can be designed to maximize the information about θ_0 available in y_t.

To formulate the input design problem, consider the FIM (recall Lemma 1.1 in page 9)

$$\mathcal{I}_F^e = \mathbf{E} \left\{ \left. \frac{\partial l_\theta(y_{1:n_{\text{seq}}})}{\partial \theta} \right|_{\theta=\theta_0} \left. \frac{\partial l_\theta(y_{1:n_{\text{seq}}})}{\partial \theta^\top} \right|_{\theta=\theta_0} \right\} \in \mathbb{R}^{n_\theta \times n_\theta}, \tag{5.4}$$

where

$$l_\theta(y_{1:n_{\text{seq}}}) = \log p_\theta(y_{1:n_{\text{seq}}}) \tag{5.5}$$

is the log-likelihood function, with $p_\theta(y_{1:n_{\text{seq}}})$ being the probability distribution of the data, $y_{1:n_{\text{seq}}}$, given the parameter θ. The FIM defines the maximal accuracy that an unbiased estimator of θ_0 can achieve, due to the Cramér-Rao bound (cf. Lemma 1.1 in page 9):

$$\text{Cov}(\hat{\theta}) \succeq \{\mathcal{I}_F^e\}^{-1}, \tag{5.6}$$

where $\hat{\theta}$ denotes the estimator of θ_0.

Given a concave matrix non-decreasing function $h : S_+^{n_\theta} \to \mathbb{R}$, where $S_+^{n_\theta}$ corresponds to the space of symmetric and positive-semidefinite matrices of dimension $n_\theta \times n_\theta$, an input design problem can be posed as

Problem 5.1 *Design an input sequence* $u^{\text{opt}}_{1:n_{\text{seq}}}$ *for the quantized system* (5.1) *such that*

$$\begin{aligned} u^{\text{opt}}_{1:n_{\text{seq}}} = \quad & \arg\max_{u_{1:n_{\text{seq}}}} \quad h(\mathcal{I}_F^e) \\ & \text{s. t.} \qquad |u_t| \leq u_{\max}, \;\; \textit{for all } t \in \{1, \ldots, n_{\text{seq}}\}\,, \end{aligned} \tag{5.7}$$

where $u_{\max} > 0$ *is a prescribed upper bound on the input amplitude.*

Problem 5.1 is difficult to solve. The main issue is that the optimization of $h(\mathcal{I}_F^e)$ as a function of $u_{1:n_{\text{seq}}}$ is typically non-convex, which gives place to numerical difficulties (e.g., local optima and initialization). However, if we relax the requirements on $u_{1:n_{\text{seq}}}$ to be a realization of a stochastic process with memory $n_m \leq n_{\text{seq}}$ and finite alphabet $\mathcal{C}$, then we can use the method described in Chapter 3 to solve a suboptimal approximation of Problem 5.1. By a suitable definition of $\mathcal{C}$, the constraint in the input design problem can be implicitly considered. However, to use the results in Chapter 3, we need an expression for $\mathcal{I}_F$ to compute by numerical simulations the required matrices to set up the optimization. The next sections are focused on providing such expression for $\mathcal{I}_F$ when the system is described according to Equation (5.1).

5.2 Background on maximum likelihood estimation with quantized output data

The maximum likelihood (ML) estimation problem can be posed as:

$$\hat{\theta}_{n_{\text{seq}}} = \arg\max_{\theta} l_\theta(y_{1:n_{\text{seq}}}), \tag{5.8}$$

where the log-likelihood function $l_\theta(y_{1:n_{\text{seq}}})$ is given by (5.5). For the problem of interest in this chapter, we can write (5.5) as

$$l_\theta(y_{1:n_{\text{seq}}}) = \sum_{t=1}^{n_{\text{seq}}} l_t(\theta), \quad l_t(\theta) = \log p_\theta(y_t|y_{1:t-1}). \tag{5.9}$$

Assumption 5.1 *The input* $u_{1:n_{\text{seq}}} = \{u_t\}_{t=1}^{n_{\text{seq}}}$ *is an exogenous signal, and the FIR system is in open loop.*

Remark 5.2 *Notice that the system (of interest) is FIR, thus,*

$$l_t(\theta) = \log p_\theta(y_t|y_{1:t-1}) = \log p_\theta(y_t)\,.$$

Before introducing the next assumption, we need the following definitions [59]:

Definition 5.1 *(Uniformly stable filters) Consider the family of filters*

$$\mathcal{H} = \{G_\beta(q) = \sum_{k=1}^{\infty} g_{\beta,\,k} q^{-k} \,|\beta \in \mathcal{B}\}\,. \tag{5.10}$$

The family of filters $\mathcal{H}$ is uniformly stable *if and only if*

$$|g_{\beta,\,k}| \leq g_k, \;\; \textit{for all } \beta \in \mathcal{B}\,, \tag{5.11}$$

where $\{g_k\}_{k=1}^{\infty}$ satisfies

$$\sum_{k=1}^{\infty} g_k < \infty\,. \tag{5.12}$$

Definition 5.2 *(Quasi-stationary signal) A signal $\{s_t\}$ is said to be* quasi-stationary *if and only if, for all integers t, r, and τ,*

$$\mathbf{E}\{s_t\} = m_s(t)\,, \tag{5.13}$$

$$\mathbf{E}\{s_t\, s_r\} = R_s(t,\, r)\,, \tag{5.14}$$

$$\lim_{n_{\rm seq}\to\infty} \frac{1}{n_{\rm seq}} \sum_{i=1}^{n_{\rm seq}} R_s(i,\, i-\tau) = R'_s(\tau)\,, \tag{5.15}$$

where the functions m_s, R_s, satisfy $|m_s(t)| \leq C$, $|R_s(t,\, r)| \leq C$, for some finite $C \geq 0$.

Definition 5.3 *(Informative enough data sets) A quasi-stationary data set $\{y_t,\, u_t\}$ is* informative enough *with respect to the model set $\mathcal{M}$ if and only if, for two optimal one-step ahead prediction filters $W_1(q)$ and $W_2(q)$, derived from any two models in the model set $\mathcal{M}$, the expression*

$$\lim_{n_{\rm seq}\to\infty} \frac{1}{n_{\rm seq}} \sum_{t=1}^{n_{\rm seq}} \mathbf{E}[(W_1(q) - W_2(q))\, z_t]^2 = 0\,, \tag{5.16}$$

implies that $W_1(e^{i\omega}) = W_2(e^{i\omega})$ for almost all ω, where $z_t := \begin{bmatrix} y_t^\top & u_t^\top \end{bmatrix}^\top$.

Assumption 5.2 *The vector of parameters θ, the input u_t and the noise w_t satisfy the following regularity conditions [59, Theorem 8.3]:*

- *The data set $\{y_t,\, u_t\}$ is such that for some filters $\{d^{(i)}_{t,\,k}\}$,*

$$\begin{aligned} x_t &= \sum_{k=1}^{\infty} d^{(1)}_{t,\,k} u_{t-k} + \sum_{k=1}^{\infty} d^{(2)}_{t,\,k} w_{t-k}, \\ y_t &= \mathcal{Q}[x_t]. \end{aligned} \tag{5.17}$$

- $\{u_t\}$ *is a bounded, external input sequence.*
- $\{w_t\}$ *is a sequence of independent random variables with zero mean values, and bounded moments of order* $4+\delta$ *for some* $\delta > 0$.
- *The family of filters* $\{d^{(i)}_{t,\,k}\}_{k=1}^{\infty}$, $i \in \{1,\,2\}$, $t \geq 1$ *is uniformly stable.*
- *The signals* $\{y_t\}$, $\{u_t\}$ *are jointly quasi-stationary.*
- *There exists a* θ_0, *such that* y_t *is described by* (5.1) *when* $\theta = \theta_0$ *(no undermodeling).*
- *The data set* $\{y_t,\,u_t\}$ *is informative enough with respect to the model structure* (5.1).

The conditions presented in Assumption 5.2 guarantee that the solution $\hat{\theta}_{n_{\text{seq}}}$ of the optimization problem in (5.8) converges (in probability or a.s.) to the true solution θ_0 as $n_{\text{seq}} \to \infty$ [58].

The maximum likelihood estimation (5.8) requires the computation of the log-likelihood function associated to a model in the set (5.3), which is difficult to handle due to the quantizer in (5.1). To overcome this issue, we use the expectation maximization (EM) algorithm to solve maximum likelihood problem (5.8). Moreover, the intermediate results of the EM algorithm are used to compute the Fisher information matrix of the system (5.1), as it is shown in Section 5.3.

The EM algorithm [20, 66] is an iterative procedure that at the k-th step seeks a value $\hat{\theta}_{n_{\text{seq}},k}$ such that the likelihood is increased in that $\log p_{\hat{\theta}_{n_{\text{seq}},k}}(y_{1:n_{\text{seq}}}) > \log p_{\hat{\theta}_{n_{\text{seq}},k-1}}(y_{1:n_{\text{seq}}})$.

The main idea of the EM algorithm is the postulation of a *missing* data set $x_{1:n_{\text{seq}}}$. In this chapter, the missing data $x_{1:n_{\text{seq}}}$ will be understood as the state sequence $\{x_t\}$ in the model structure (5.1), but other choices are possible, and it can be considered as a design variable. This approach assumes that maximizing the joint log-likelihood $\log p_\theta(x_{1:n_{\text{seq}}},\, y_{1:n_{\text{seq}}})$ is easier than maximizing the marginal one $\log p_\theta(y_{1:n_{\text{seq}}})$. We refer to Appendix C for more details about the EM algorithm.

The EM algorithm consists of two steps: (i) the calculation of an auxiliary function (E-step) and, (ii) the optimization of this auxiliary function (M-step). The E-step is obtained by calculating the function $\mathbf{Q}(\theta, \hat{\theta}_{n_{\text{seq}},i})$:

$$\mathbf{Q}(\theta, \hat{\theta}_{n_{\text{seq}},i}) := \int_{\mathbb{R}^{n_{\text{seq}}}} \log[p_\theta(x_{1:n_{\text{seq}}}, y_{1:n_{\text{seq}}})]\, p_{\hat{\theta}_{n_{\text{seq}},i}}(x_{1:n_{\text{seq}}} | y_{1:n_{\text{seq}}}) dx_{1:n_{\text{seq}}}. \quad (5.18)$$

The M-step is then given by

$$\hat{\theta}_{n_{\text{seq}},i+1} = \arg\max_{\theta} \mathbf{Q}(\theta, \hat{\theta}_{n_{\text{seq}},i}). \quad (5.19)$$

Before introducing one of the main results in this section, we need the following lemma [31, Lemma 5]:

Lemma 5.1 *The joint log-likelihood* $\log p_\theta(x_{1:n_{\text{seq}}}, y_{1:n_{\text{seq}}})$*, and the conditional pdf*

$$p_{\hat{\theta}_{n_{\text{seq}},i}}(x_{1:n_{\text{seq}}}|y_{1:n_{\text{seq}}})$$

are given by

$$\log p_\theta(x_{1:n_{\text{seq}}}, y_{1:n_{\text{seq}}}) = \begin{cases} \sum_{t=1}^{n_{\text{seq}}} \log p_\theta(x_t), & x_t \in \mathcal{Q}^{-1}[y_t]\,, \\ 0, & \textit{otherwise,} \end{cases} \tag{5.20}$$

$$p_{\hat{\theta}_{n_{\text{seq}},i}}(x_{1:n_{\text{seq}}}|y_{1:n_{\text{seq}}}) = \prod_{t=1}^{n_{\text{seq}}} p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)\,, \tag{5.21}$$

with $p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)$ *given by*

$$p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t) = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{\mathcal{S}_t} \exp\left\{-\frac{1}{2\sigma^2}(x_t - \varphi_t^\top \theta)^2\right\} dx_t\,. \tag{5.22}$$

Also, the E-step in the EM algorithm is given by

$$\mathbf{Q}(\theta,\, \hat{\theta}_{n_{\text{seq}},i}) = -\frac{1}{2}\sum_{t=1}^{n_{\text{seq}}} \left[\log\{2\pi\sigma^2\} + \frac{1}{\sigma^2}\int_{\mathcal{S}_t} (x_t - \varphi_t^\top\theta)^2 p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t) dx_t\right], \tag{5.23}$$

where $\mathcal{S}_t = \{\tilde{x}_t : \mathcal{Q}[x_t] = y_t\}$.

Proof *The expression of the joint log-likelihood can be obtained as*

$$\begin{aligned} p_\theta(x_{1:n_{\text{seq}}}, y_{1:n_{\text{seq}}}) &= \prod_{t=1}^{n_{\text{seq}}} p_\theta(x_t,\, y_t|x_{1:t-1}, y_{1:t-1}) \\ &= \prod_{t=1}^{n_{\text{seq}}} p_\theta(x_t,\, y_t)\,, \end{aligned} \tag{5.24}$$

with $p_\theta(x_t,\, y_t)$ *defined by*

$$p_\theta(x_t,\, y_t) = \begin{cases} p_\theta(x_t), & x_t \in \mathcal{Q}^{-1}[y_t]\,, \\ 0, & \textit{otherwise,} \end{cases} \tag{5.25}$$

and from [33, p. 210] we know that

$$x_t \sim \mathcal{N}(\varphi_t^\top\theta,\, \sigma^2)\,. \tag{5.26}$$

Hence,

$$\log p_\theta(x_{1:n_{\text{seq}}}, y_{1:n_{\text{seq}}}) = \sum_{t=1}^{n_{\text{seq}}} \log p_\theta(x_t,\, y_t)\,, \tag{5.27}$$

and (5.20) *follows from* (5.25). *Also, from* (5.24) *and*

$$p_\theta(y_{1:n_{\text{seq}}}) = \prod_{t=1}^{n_{\text{seq}}} p_\theta(y_t|y_{1:t-1}) = \prod_{t=1}^{n_{\text{seq}}} p_\theta(y_t)\,, \tag{5.28}$$

we have that

$$\begin{aligned} p_{\hat{\theta}_{n_{\text{seq}},i}}(x_{1:n_{\text{seq}}}|y_{1:n_{\text{seq}}}) &= \frac{p_{\hat{\theta}_{n_{\text{seq}},i}}(x_{1:n_{\text{seq}}},\, y_{1:n_{\text{seq}}})}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_{1:n_{\text{seq}}})} \\ &= \prod_{t=1}^{n_{\text{seq}}} \frac{p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t,\, y_t)}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t)} = \prod_{t=1}^{n_{\text{seq}}} p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)\,. \end{aligned} \tag{5.29}$$

In the view of (5.18)*, to compute the E-step we need the expressions for the joint log-likelihood* $\log p_\theta(x_{1:n_{\text{seq}}}, y_{1:n_{\text{seq}}})$*, and the conditional pdf* $p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)$*. Combining* (5.18) *with* (5.27) *and* (5.29) *we have that*

$$\mathbf{Q}(\theta, \hat{\theta}_{n_{\text{seq}},i}) = \int_{\mathbb{R}^{n_{\text{seq}}}} \sum_{t=1}^{n_{\text{seq}}} \log[p_\theta(x_t, y_t)] \prod_{s=1}^{n_{\text{seq}}} p_{\hat{\theta}_{n_{\text{seq}},i}}(x_s|y_s) dx_{1:n_{\text{seq}}} \tag{5.30}$$

$$= \sum_{t=1}^{n_{\text{seq}}} \int_{\mathbb{R}} \log[p_\theta(x_t, y_t)]\, p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t) dx_t\,, \tag{5.31}$$

and (5.23) *follows from* (5.26) *and* (5.25). ■

In [31], it was shown that under the stated assumptions, a parameter estimate $\hat{\theta}_{n_{\text{seq}}}$ can be found using the EM algorithm. This algorithm is summarized in the following theorem:

Theorem 5.1 *Consider the system given in* (5.1)*, Assumptions 5.1 and 5.2, and the maximization problem stated in* (5.19)*. Then the M-step of the EM algorithm is given by:*

$$\hat{\theta}_{n_{\text{seq}},i+1} = \left[\sum_{t=1}^{N} \varphi_t \varphi_t^\top\right]^{-1} \sum_{t=1}^{N} \varphi_t \hat{x}_t, \tag{5.32}$$

where

$$\hat{x}_t = \varphi_t^\top \hat{\theta}_{n_{\text{seq}},i} + \sigma \frac{I_t^{(1)}}{I_t^{(0)}}, \tag{5.33}$$

with $I_t^{(0)} \in \mathbb{R}$*,* $I_t^{(1)} \in \mathbb{R}$ *given by*

$$I_t^{(0)} = \frac{1}{\sqrt{2\pi}} \int_{\tilde{\mathcal{S}}_t} \exp\left\{-\frac{1}{2}\tilde{x}_t^2\right\} d\tilde{x}_t, \tag{5.34}$$

$$I_t^{(1)} = \frac{1}{\sqrt{2\pi}} \int_{\tilde{\mathcal{S}}_t} \tilde{x}_t \exp\left\{-\frac{1}{2}\tilde{x}_t^2\right\} d\tilde{x}_t, \tag{5.35}$$

and $\tilde{\mathcal{S}}_t = \{\tilde{x}_t : \mathcal{Q}[\sigma\tilde{x}_t + \varphi_t^\top \hat{\theta}_{n_{\text{seq}},i}] = y_t\}$.

Proof *The maximization of* (5.19) *with respect to* θ *can be found by taking the derivative in* (5.23) *with respect to* θ*, and using the definition for* $\mathcal{S}_t$*. Thus,*

$$\begin{aligned}
-2\frac{\partial \mathbf{Q}(\theta, \hat{\theta}_{n_{\text{seq}},i})}{\partial \theta} &= \frac{\partial}{\partial \theta}\left(\sum_{t=1}^{n_{\text{seq}}}\left[\log\{2\pi\sigma^2\} + \frac{1}{\sigma^2}\int_{\mathcal{S}_t}(x_t - \varphi_t^\top\theta)^2 p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)dx_t\right]\right) \\
&= \sum_{t=1}^{n_{\text{seq}}}\left[\frac{1}{\sigma^2}\int_{\mathcal{S}_t}\frac{\partial}{\partial\theta}(x_t - \varphi_t^\top\theta)^2 p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)dx_t\right] \\
&= \sum_{t=1}^{n_{\text{seq}}}\left[-\frac{2}{\sigma^2}\int_{\mathcal{S}_t}\varphi_t(x_t - \varphi_t^\top\theta) p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)dx_t\right]. \qquad (5.36)
\end{aligned}$$

Setting (5.36) *equal to zero we have that*

$$\hat{\theta}_{n_{\text{seq}},i+1} = \left[\sum_{t=1}^{n_{\text{seq}}}\varphi_t\varphi_t^\top\right]^{-1}\sum_{t=1}^{n_{\text{seq}}}\varphi_t\hat{x}_t\,, \qquad (5.37)$$

where

$$\hat{x}_t = \int_{\mathcal{S}_t} x_t\, p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t|y_t)\, dx_t\,. \qquad (5.38)$$

From (5.38) *we have*

$$\begin{aligned}
\hat{x}_t &= \int_{\mathbb{R}} x_t \frac{p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t,\, y_t)}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t)}\, dx_t = \frac{1}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t)}\int_{\mathcal{S}_t} x_t p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t)\, dx_t \\
&= \frac{1}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t)}\int_{\mathcal{S}_t}\frac{x_t}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{1}{2\sigma^2}(x_t - \varphi_t^\top\hat{\theta}_{n_{\text{seq}},i})^2\right\} dx_t\,. \qquad (5.39)
\end{aligned}$$

Making the change of variables $\tilde{x}_t = \sigma^{-1}(x_t - \varphi_t^\top\hat{\theta}_{n_{\text{seq}},i})$*, we have*

$$\begin{aligned}
\hat{x}_t &= \frac{1}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t)}\int_{\tilde{\mathcal{S}}_t}\frac{(\sigma\tilde{x}_t + \varphi_t^\top\hat{\theta}_{n_{\text{seq}},i})}{\sqrt{2\pi\sigma^2}}e^{-\frac{1}{2}\tilde{x}_t^2}\, d\tilde{x}_t \\
&= \frac{1}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t)}\left[\int_{\tilde{\mathcal{S}}_t}\frac{\sigma\tilde{x}_t}{\sqrt{2\pi\sigma^2}}e^{-\frac{1}{2}\tilde{x}_t^2}\, d\tilde{x}_t + \varphi_t^\top\hat{\theta}_{n_{\text{seq}},i}\int_{\tilde{\mathcal{S}}_t}\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{1}{2}\tilde{x}_t^2}\, d\tilde{x}_t\right] \\
&= \frac{1}{p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t)}\left[\sigma I_t^{(1)} + \varphi_t^\top\hat{\theta}_{n_{\text{seq}},i} I_t^{(0)}\right]. \qquad (5.40)
\end{aligned}$$

Also,

$$\begin{aligned}
p_{\hat{\theta}_{n_{\text{seq}},i}}(y_t) &= \int_{\mathbb{R}} p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t,\, y_t)\, dx_t = \int_{\mathcal{S}_t} p_{\hat{\theta}_{n_{\text{seq}},i}}(x_t)\, dx_t \\
&= \int_{\tilde{\mathcal{S}}_t}\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{1}{2}\tilde{x}_t^2\right\}\sigma\, d\tilde{x}_t
\end{aligned}$$

$$= I_t^{(0)} \,. \tag{5.41}$$

Finally, the expression for $\hat{x}_t$ *follows from* (5.40) *and* (5.41). ■

Developing further the expressions for the integrals $I_t^{(0)}$, $I_t^{(1)}$, we can obtain:

$$I_t^{(0)} = 0.5 \ \mathrm{erf}(\tilde{\mathcal{S}}_t/\sqrt{2}), \quad I_t^{(1)} = -\frac{e^{-x^2/2}}{\sqrt{2\pi}}\bigg|_{x\in\tilde{\mathcal{S}}_t}, \tag{5.42}$$

where $f(x)|_{x\in\tilde{\mathcal{S}}_t}$ denotes the function f evaluated at the maximum and minimum values of the interval $\tilde{\mathcal{S}}_t = \{\tilde{x}_t : \mathcal{Q}[\sigma\tilde{x}_t + \varphi_t^\top\theta] = y_t\}$, and

$$\mathrm{erf}(\mathcal{A}) := \frac{2}{\sqrt{\pi}}\int_{\mathcal{A}} e^{-u^2} du$$

is a modified error function, defined for sets.

5.3 Fisher information matrix for systems with quantized output

To use the technique developed in this thesis, we need an expression for the Fisher information matrix $\mathcal{I}_F$. Before introducing the main result, we need the following lemma [31]:

Lemma 5.2 *The partial derivative of* $\mathbf{Q}(\theta,\,\theta_i)$ *with respect to* θ *can be expressed as*

$$\frac{\partial\mathbf{Q}(\theta,\,\theta_i)}{\partial\theta} = \sum_{t=1}^{n_{\mathrm{seq}}}\left(\frac{1}{\sigma^2}\varphi_t\varphi_t^\top[\theta_i - \theta] + \frac{1}{\sigma}\frac{I_t^{(1)}}{I_t^{(0)}}\right). \tag{5.43}$$

Proof *We start from* (5.36) *to write*

$$\begin{aligned}\frac{\partial\mathbf{Q}(\theta,\,\theta_i)}{\partial\theta} &= \sum_{t=1}^{n_{\mathrm{seq}}}\left[\frac{1}{\sigma^2}\int_{\mathcal{S}_t}\varphi_t(x_t - \varphi_t^\top\theta)p_{\hat{\theta}_{n_{\mathrm{seq}},i}}(x_t|y_t)dx_t\right]\\ &= \sum_{t=1}^{n_{\mathrm{seq}}}\frac{\varphi_t}{\sigma^2}\left[\hat{x}_t - \varphi_t^\top\theta\int_{\mathcal{S}_t}p_{\theta_i}(x_t|y_t)dx_t\right],\end{aligned} \tag{5.44}$$

where $\hat{x}_t$ *is defined according to Theorem 5.1. Moreover, since* $p_{\theta_i}(x_t|y_t)$ *is nonzero only in* $\mathcal{S}_t$*, we have*

$$\int_{\mathcal{S}_t} p_{\theta_i}(x_t|y_t)dx_t = 1\,. \tag{5.45}$$

Finally, using (5.33) *and* (5.45) *into* (5.44) *we obtain* (5.43). ■

The expression for $\mathcal{I}_F$ for the system described in equation (5.1) is provided in the following lemma:

Lemma 5.3 *(Fisher information matrix for models with quantized output) For systems with quantized output data of the form* (5.1)*, the FIM is given a.s. by the following expression:*

$$\mathcal{I}_F = \lim_{N_{\text{sim}}\to\infty} \frac{2}{\pi\sigma^2 N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \left(\frac{e^{-x^2/2}\big|_{x\in\tilde{\mathcal{S}}_t}}{\operatorname{erf}(\tilde{\mathcal{S}}_t/\sqrt{2})} \right)^2 \varphi_t\varphi_t^\top \ , \ \textit{a.s.} \ , \tag{5.46}$$

where $f(x)|_{x\in\tilde{\mathcal{S}}_t}$ *denotes the function* f *evaluated at the maximum and minimum values of the interval* $\tilde{\mathcal{S}}_t = \{\tilde{x}_t : \mathcal{Q}[\sigma\tilde{x}_t + \varphi_t^\top\theta] = y_t\}$.

Proof *We start by using the Fisher's identity*[1] *[66, p.80]*

$$\left.\frac{\partial l_\theta(y_{1:n_{\text{seq}}})}{\partial\theta}\right|_{\theta=\theta_0} = \left.\frac{\partial \mathbf{Q}(\theta,\theta_0)}{\partial\theta}\right|_{\theta=\theta_0} \tag{5.47}$$

where $\mathbf{Q}(\theta,\theta_0)$ *is the auxiliary function arising from the EM algorithm.*

From Lemma 5.2 we have that

$$\left.\frac{\partial \mathbf{Q}(\theta,\theta_0)}{\partial\theta}\right|_{\theta=\theta_0} = \frac{1}{\sigma}\sum_{t=1}^{n_{\text{seq}}} \varphi_t \frac{I_t^{(1)}}{I_t^{(0)}}. \tag{5.48}$$

For the SISO case, the quotient $I_t^{(1)}/I_t^{(0)}$ *can be expressed as:*

$$\frac{I_t^{(1)}}{I_t^{(0)}} = -\frac{\sqrt{2}}{\sqrt{\pi}} \frac{e^{-x^2/2}\Big|_{x\in\tilde{\mathcal{S}}_t}}{\operatorname{erf}(\tilde{\mathcal{S}}_t/\sqrt{2})}. \tag{5.49}$$

Thus, we can write the FIM as

$$\begin{aligned}\mathcal{I}_F &= \mathbf{E}\left\{ \left.\frac{\partial l_\theta(y_{1:n_{\text{seq}}})}{\partial\theta}\right|_{\theta=\theta_0} \left.\frac{\partial l_\theta(y_{1:n_{\text{seq}}})}{\partial\theta^\top}\right|_{\theta=\theta_0} \right\} \\ &= \frac{2}{\pi\sigma^2}\mathbf{E}\left\{ \left(\frac{e^{-x^2/2}\big|_{x\in\tilde{\mathcal{S}}_t}}{\operatorname{erf}(\tilde{\mathcal{S}}_t/\sqrt{2})} \right)^2 \varphi_t\varphi_t^\top \right\}. \end{aligned} \tag{5.50}$$

Since x_t *and* φ_t *are asymptotically jointly stationary processes, asymptotically uncorrelated (for sufficiently large lags), by the Birkhoff-Khinchin ergodic theorem [79, Lemma B.1], we can write* (5.50) *as:*

$$\mathcal{I}_F = \lim_{N_{\text{sim}}\to\infty} \frac{2}{\pi\sigma^2 N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \left(\frac{e^{-x^2/2}\big|_{\tilde{\mathcal{S}}_t}}{\operatorname{erf}(x/\sqrt{2})\big|_{\tilde{\mathcal{S}}_t}} \right)^2 \varphi_t\varphi_t^\top \ , \ \textit{a.s.} \tag{5.51}$$

which is the expression in (5.46). ■

[1] We refer to Appendix C, page 126 for the derivation of Fisher's identity.

Remark 5.3 *Notice that a more general case can be developed considering incomplete data (e.g., the output y_t is missed for some t). In particular, missing data is a common type of incomplete data, and to calculate the FIM we still can use the Fisher's identity, but we need to consider that the expectations in* (5.18) *are now taken with respect to a different set of observations. However, one important assumption to consider is that the missing data mechanism is somehow independent of the stochastic processes in the system, see e.g. [1].*

Lemma 5.3 provides a numerical approximation of $\mathcal{I}_F$, which is based on the results introduced in [31]. Therefore, Lemma 5.3 can be employed to approximate numerically the Fisher information matrices associated with the set of extreme points of stationary processes with memory n_m and alphabet $\mathcal{C}$, which are employed to solve Problem 5.1 in page 80. To this end, we use the steps for the input design method presented in page 43.

5.4 Numerical example

To illustrate the results in this chapter, we consider the following example:

Example 5.1 *Consider a SISO FIR system, defined as*

$$\begin{aligned} x_t &= \varphi_t^\top \theta_0 + w_t, \\ y_t &= \mathcal{Q}[x_t], \end{aligned} \tag{5.52}$$

where $\mathcal{Q}$ is a 2-bit quantizer, $\varphi_t = \begin{bmatrix} u_{t-1} & u_{t-2} \end{bmatrix}^\top$, $\theta_0 = \begin{bmatrix} 0.4 & 1 \end{bmatrix}^\top$, and the noise variance is $\sigma^2 = 1$. The extreme values for $\mathcal{Q}$ are -7.5 and 7.5, with equally spaced output values on $[-7.5, 7.5]$. Thus, the quantizer is given by

$$\mathcal{Q}[x] = \begin{cases} 7.5, & \text{if } x > 2.5, \\ 2.5, & \text{if } x \in (0, 2.5], \\ -2.5, & \text{if } x \in [-2.5, 0], \\ -7.5, & \text{if } x < -2.5. \end{cases} \tag{5.53}$$

We use the method presented in Section 3.2 in page 42 to design an experiment for identifying θ_0 in (5.52)*. For this purpose, we consider an input with maximum amplitude 5, and different scenarios, composed by memory length $n_m \in \{2, 3\}$, and $n_{\mathcal{C}} \in \{2, 4\}$. We solve* (3.19) *for $h(\cdot) = -tr\{W(\cdot)^{-1}\}$, where*

$$W = \begin{bmatrix} 4 & 3 \\ 3 & 4 \end{bmatrix}. \tag{5.54}$$

The approximation of each $\mathcal{I}_F^{(i)}$ in (3.20a) *is obtained using an approximation of* (5.46) *with $N_{\text{sim}} = 5 \cdot 10^5$. The problem is solved in* `Matlab` *using the* `cvx` *toolbox [35].*

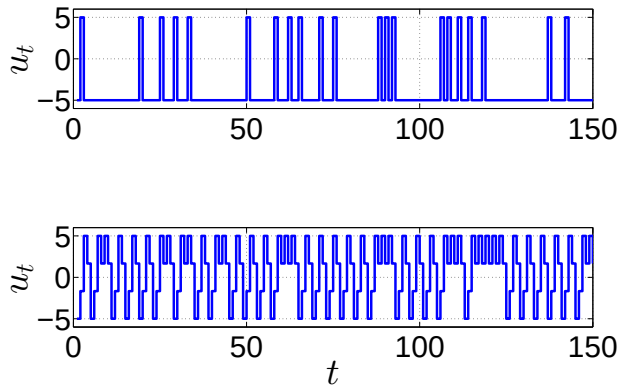

Figure 5.2: Input realizations for memory length $n_m = 2$. Top: $n_{\mathcal{C}} = 2$. Bottom: $n_{\mathcal{C}} = 4$.

Table 5.1: $h(\mathcal{I}_F)$ for different values of $n_{\mathcal{C}}$ and memory length (of the Markov chain input).

memory length $\diagdown$ $n_{\mathcal{C}}$	2	4
2	−3.12	−1.58
3	−3.08	−1.58

Once the problem (3.19) *is solved, we generate an input signal of length* $n_{\text{seq}} = 10^6$ *from the optimal pmf by running a Markov chain based on* (2.27)*. Typical input realizations for memory length* 2 *are presented in Figure 5.2. To compare the results, we compute* $h(\mathcal{I}_F)$ *for each optimal input signal using* (5.46)*.*

The results obtained for the different scenarios are presented in Table 5.1. The results show that the value of h *is increased when we increase the number of possible values for the input. However, there is no significant improvement when we increase the memory length of the stationary process, which says that a memory length equal to* n_θ *might be optimal for this example.*

As a benchmark, we generate a binary white noise input of length n_{seq} *and amplitude* 5*, and we compute* $h(\mathcal{I}_F)$ *for this input using* (5.46)*. The value we obtain is* $h(\mathcal{I}_F) = -3.75$*, which is 16.8% worse than the worst result presented in Table 2.1. Therefore, for this example the proposed input design method outperforms the result obtained with binary white noise.*

5.5 Conclusion

In this chapter we introduce an application of the input design method presented in previous chapters. The goal is to design an input sequence as a realization of a stationary process, maximizing the information retrieved from the experiment resulting of applying the input sequence to a system with quantized output. The main difficulty when using the proposed technique on these systems is that the computation of the Fisher information matrix becomes difficult since the output measurements are quantized. To solve this issue, this chapter provides an estimate of the Fisher information matrix based on Fisher's identity. The estimate is computed by numerical approximations, which are employed to estimate the Fisher information matrices associated with the probability measures describing the set of stationary processes. The numerical example illustrates that the proposed method can be employed to design input sequences for systems with quantized output. Future work in the subject will consider more complex model structures, which implies that new expressions to approximate the Fisher information matrix will be needed.

Chapter 6

Application oriented closed-loop input design

In previous chapters we have discussed the problem of input design for identification of systems in open-loop operation. However, in many industrial applications the data collected for identification comes from systems working in closed-loop [59, 89, 92].

In practical applications, many systems can only work in closed-loop settings due to stability issues, production restrictions, economic considerations or inherent feedback mechanisms. On the other hand, it is sometimes required to update the existing control laws or design a new controller. Since most of the methods for designing controllers require the knowledge of the system to be controlled, closed-loop system identification is a building block in this process. The main burden in closed-loop identification is the correlation between the measurement noise and input signal, which is imposed on the experiment by the feedback loop. There exists quite a rich literature on closed-loop identification with three main approaches: direct methods (the model is identified as if the system were operating in open-loop), indirect methods (the model is obtained from the identified closed-loop structure), and joint input-output methods (an augmented model is identified, where the input and output of the system are considered as the new outputs, and the reference and noise as new inputs), see, e.g., [27, 59, 79, 91] and the references therein.

One crucial question that arises in any identification process is how to generate data efficiently. This question is addressed by input design methods, where the objective is to generate an input signal that maximizes the information retrieved from an experiment [3, 33]. In this area, application oriented input design is one approach to formulate the optimal input design problem. The main idea is to guarantee a certain control performance for the identified model with the least possible experimental effort. The same idea has been used in identification for control and least costly identification, see, e.g., [3, 30, 39, 52, 53, 55].

For closed-loop models, the input design problem is often translated to the

design of the spectrum of an additive external excitation. There exists a vast literature on closed-loop experiment design, where the controller can also be designed (provided it is a design variable). In this line, [37] designs an identification experiment minimizing the uncertainty region for the identified model. The experiment design is performed in the frequency domain, where the input spectrum is design to minimize the uncertainty region for the identified model, while satisfying power contraints. In [38] the design of an optimal experiment for parametric prediction error system identification of linear time-invariant systems is considered, where the system is assumed to operate in closed loop. To this end, [38] parameterizes the set of admissible controller-external input pairs by a finite set of matrix valued trigonometric moments. In [41, 44] the problem of experiment design for closed loop system identification is analyzed. In this case, [41, 44] use a finite dimensional parameterization of the input spectrum and the Youla-Kucera parameterization to recast the problem as a semidefinite program. However, the main limitation on the existing methods is that they cannot be employed in closed-loop systems with nonlinear feedback. In addition, they cannot handle probabilistic constraints on the input and output, which arise for safety or practical limitations.

In this chapter we present a new approach for application oriented experiment design for closed-loop systems. We consider a linear time-invariant system being controlled by a known controller (either linear or nonlinear), where the main goal is reference tracking. Due to a performance degradation (e.g., a change in the process dynamics producing a degradation in the quality of reference tracking), we want to update the current controller or design another one, and thus a plant model needs to be identified. Since the controller is known we will employ indirect identification, where the model is identified by adding an external stationary input. The input design is then formulated as an optimization problem, where we design the external excitation achieving minimum experimental effort, while we are also taking care of the tracking performance of the existing controller.

The designed input sequence must guarantee that the estimated parameters satisfies a prescribed accuracy. Since the covariance matrix of the asymptotic distribution of the parameter estimates depends on the Fisher information matrix, it is reasonable to impose a constraint as a function of this matrix to guarantee a prescribed accuracy in the estimated parameters. Thus, to guarantee that the model parameters are estimated with the desired accuracy, we add a constraint on the quality of the estimated model in terms of the Fisher information matrix [59], to obtain an input signal guaranteeing that the estimated model is in the set of models that satisfy the desired control specifications, with a given probability.

In practice we also have bounds on the input and output signals, which should be taken into account during the experiment design. However, the measured output of the system is usually corrupted by an unknown disturbance, which is modeled as stochastic process. Moreover, since the system is operating in closed loop, the input of the system will be also affected by the unknown disturbance. Therefore, any bounds on the input and output must be given in terms of probability measures. Consequently, the proposed optimization also considers probabilistic bounds for the

input and output of the system.

The obtained optimization problem is nonconvex due to the constraints, and thus it is difficult to handle. This issue is relaxed by extending the method introduced in this thesis and [87] to closed-loop and constrained system identification. By assuming that the external excitation is a realization from a stationary process with finite memory and finite alphabet, we can characterize the stationary process describing the external excitation as a convex combination of the basis inputs. Since the basis inputs are known, we can estimate the Fisher information matrix, the probabilities, and the cost function associated with each basis input by using numerical methods. Therefore, the resulting optimization problem needs to find the optimal convex combination of the basis inputs minimizing the cost function, while satisfying the probabilistic and accuracy constraints. Consequently, the optimization problem is convex in the decision variables, which makes it tractable.

6.1 Preliminaries

Consider the discrete time, linear, time-invariant system

$$\begin{aligned} x_{t+1} &= A(\theta_0)x_t + B(\theta_0)u_t, \\ y_t &= C(\theta_0)x_t + \nu_t. \end{aligned} \tag{6.1}$$

where $u_t \in \mathbb{R}^{n_u}$ and $y_t \in \mathbb{R}^{n_y}$ are the input and output vectors. $\nu_t \in \mathbb{R}^{n_e}$ is a coloured noise sequence with $\nu_t = H(q;\theta_0)e_t$, where H is a rational noise filter in terms of the time shift operator q, and $\{e_t\}$ is white noise sequence with zero mean and covariance matrix $\Lambda_e \in \mathbb{R}^{n_e \times n_e}$. In addition, we assume that H is stable, inversely stable, and satisfies $\lim_{q\to\infty} H(q;\theta) = I$.

System identification

In system identification, we aim to find a model of the system (6.1). The model is parameterized by an unknown parameter vector $\theta \in \mathbb{R}^{n_\theta}$, that is,

$$\begin{aligned} x_{t+1} &= A(\theta)x_t + B(\theta)u_t\,, \\ y_t &= C(\theta)x_t + \nu_t\,, \end{aligned} \tag{6.2}$$

where $\nu_t = H(q;\theta)e_t$. The model coincides with (6.1) when $\theta = \theta_0$ [59]. To obtain the desired results, we assume the following:

Assumption 6.1 *The model structure* (6.2) *is globally identifiable at* θ_0.

We employ the prediction error method (PEM) with quadratic cost to calculate an approximation of the unknown parameters $\theta \in \mathbb{R}^{n_\theta}$, based on N available samples of input-output, i.e., data $\{u_t, y_t, t = 1,\dots,N\}$ (cf. Chapter 1). An important asymptotic (in the sample size N) property of PEM, is that

$$\sqrt{N}(\hat{\theta}_N - \theta_0) \in \text{As}\mathcal{N}(0,\, \{\mathcal{I}_F^e\}^{-1}), \tag{6.3}$$

where $\mathcal{I}_F^e$ quantifies the information regarding the unknown parameters in the observations of the output signal (cf. Lemma 1.1 in page 9). Thus, for sufficiently large samples N, we get that with a certain probability α the estimated parameters belong to an *identification set* (see [61]) defined as

$$\Theta_{\mathrm{SI}}(\alpha) = \left\{\theta : (\theta - \theta_0)^\top \mathcal{I}_F^e (\theta - \theta_0) \leq \chi_\alpha^2(n_\theta)\right\}, \tag{6.4}$$

where $\chi_\alpha^2(n)$ is the α-percentile of the χ^2-distribution with n degrees of freedom, which in turn implies that $\hat{\theta}_N \in \Theta_{\mathrm{SI}}(\alpha)$ with probability α for sufficiently large samples N.

Application oriented input design

In application oriented input design, the main focus is to design an input signal to be used in an identification experiment such that an acceptable control performance can be guaranteed when the estimated model is used in the control design. This requires that $\hat{\theta}_N \in \Theta(\gamma)$ with high probability, where $\Theta(\gamma)$, also known as *application set*, is the set of all acceptable parameters from a control's point of view, and γ is a user-defined positive constant which imposes an upper bound on the performance degradation. One way to ensure this is to require

$$\Theta_{\mathrm{SI}}(\alpha) \subseteq \Theta(\gamma). \tag{6.5}$$

Using (6.5), the input design problem can be formulated as a constrained optimization problem with (6.5) as the constraint. Thus, a natural objective in the input design is to minimize an experimental cost, such as input power or energy, or experimental time, while (6.5) is fulfilled, i.e.,

$$\begin{aligned} \min_{\text{input}} \quad & \text{Experimental Cost} \\ \text{s.t.} \quad & \Theta_{\mathrm{SI}}(\alpha) \subseteq \Theta(\gamma). \end{aligned} \tag{6.6}$$

In order to relate the control performance degradation to the plant-model mismatch, we use the concept of application cost function, where a scalar function of θ is considered as the application cost, denoted by $V_{\mathrm{app}}(\theta)$. We choose the cost function such that its minimum value occurs at $\theta = \theta_0$. In particular, if $V_{\mathrm{app}}(\theta)$ is twice differentiable in a neighbourhood of θ_0, we assume without loss of generality:

$$V_{\mathrm{app}}(\theta_0) = 0, \quad \nabla_\theta V_{\mathrm{app}}(\theta)\big|_{\theta=\theta_0} = 0 \quad \text{and} \quad \nabla_\theta^2 V_{\mathrm{app}}(\theta)\big|_{\theta=\theta_0} \succeq 0.$$

We notice that the previous properties are always possible. Indeed, if $V_{\mathrm{app}}(\theta_0) \neq 0$, then it is possible to define a new cost function $V'_{\mathrm{app}}(\theta) := V_{\mathrm{app}}(\theta) - V_{\mathrm{app}}(\theta_0)$, where $V'_{\mathrm{app}}(\theta)$ satisfies the expressions given above.

There are many possible choices of application cost functions with these properties, see, e.g., [52]. The set of all acceptable parameters, namely the application

set, is defined as

$$\Theta(\gamma) = \left\{ \theta : V_{\text{app}}(\theta) \leq \frac{1}{\gamma} \right\}. \tag{6.7}$$

To proceed, we employ the following local approximation of $\Theta(\gamma)$ invoking the Taylor expansion of $V_{\text{app}}(\theta)$ around θ_0:

$$\begin{aligned} V_{\text{app}}(\theta) &\approx V_{\text{app}}(\theta_0) + \nabla_\theta V_{\text{app}}(\theta)|_{\theta=\theta_0}^\top [\theta - \theta_0] \\ &\quad + 0.5[\theta - \theta_0]^\top \nabla_\theta^2 V_{\text{app}}(\theta)\big|_{\theta=\theta_0} (\theta_0)[\theta - \theta_0] \\ &= 0 + 0 + 0.5[\theta - \theta_0]^\top \nabla_\theta^2 V_{\text{app}}(\theta)\big|_{\theta=\theta_0} [\theta - \theta_0]. \end{aligned} \tag{6.8}$$

Thus we have the following ellipsoidal approximation of the application set (see [40]):

$$\Theta_{\text{app}}(\gamma) := \left\{ \theta : (\theta - \theta_0)^\top \nabla_\theta^2 V_{\text{app}}(\theta)\big|_{\theta=\theta_0} (\theta - \theta_0) \leq \frac{2}{\gamma} \right\}. \tag{6.9}$$

Using (6.4) and (6.9), we have that $\Theta_{\text{SI}}(\alpha) \subseteq \Theta_{\text{app}}(\gamma)$ if and only if

$$\frac{1}{\chi^2_\alpha(n_\theta)} (\theta - \theta_0)^\top \mathcal{I}_F (\theta - \theta_0) \leq \frac{\gamma}{2} (\theta - \theta_0)^\top \nabla_\theta^2 V_{\text{app}}(\theta_0)(\theta - \theta_0), \tag{6.10}$$

for all $\theta \in \Theta_{\text{SI}}(\alpha)$. Indeed, assume first that $\Theta_{\text{SI}}(\alpha) \subseteq \Theta_{\text{app}}(\gamma)$ holds. Then for all $\theta \in \Theta_{\text{SI}}(\alpha)$ we have that

$$\frac{1}{\chi^2_\alpha(n_\theta)} (\theta - \theta_0)^\top \mathcal{I}_F (\theta - \theta_0) \leq 1, \tag{6.11}$$

$$\frac{\gamma}{2} (\theta - \theta_0)^\top \nabla_\theta^2 V_{\text{app}}(\theta_0)(\theta - \theta_0) \leq 1. \tag{6.12}$$

Since $\Theta_{\text{SI}}(\alpha) \subseteq \Theta_{\text{app}}(\gamma)$, then the only option to satisfy simultaneously equations (6.11) and (6.12) is to satisfy inequality (6.10) for all $\theta \in \Theta_{\text{SI}}(\alpha)$. Conversely, if we assume that (6.10) holds for all $\theta \in \Theta_{\text{SI}}(\alpha)$, and if we consider that (6.12) holds for all $\theta \in \Theta_{\text{SI}}(\alpha)$, then (6.11) will also be satisfied for all $\theta \in \Theta_{\text{SI}}(\alpha)$, which implies that $\Theta_{\text{SI}}(\alpha) \subseteq \Theta_{\text{app}}(\gamma)$.

Equation (6.10) implies that

$$\frac{1}{\chi^2_\alpha(n_\theta)} \mathcal{I}_F \succeq \frac{\gamma}{2} \nabla_\theta^2 V_{\text{app}}(\theta)\big|_{\theta=\theta_0}. \tag{6.13}$$

Therefore, a relaxation of the optimal input design problem (6.6) can be written as

$$\begin{aligned} \min_{\text{input}} \quad & \text{Experimental Cost} \\ \text{s.t.} \quad & \frac{1}{\chi^2_\alpha(n_\theta)} \mathcal{I}_F \succeq \frac{\gamma}{2} \nabla_\theta^2 V_{\text{app}}(\theta)\big|_{\theta=\theta_0}. \end{aligned} \tag{6.14}$$

For more details on application oriented input design refer to [40].

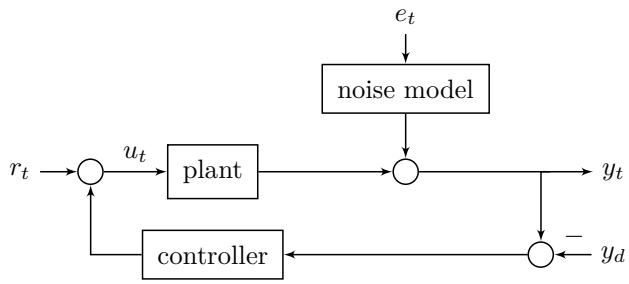

Figure 6.1: A block diagram representation of the closed-loop system.

Optimal input design via graph theory

As discussed in Chapters 3 and 4, in input design via graph theory the optimal external signal $r_{1:n_{\text{seq}}} = \{r_t\}_{t=1}^{n_{\text{seq}}}$ is designed as a realization of a stationary process with finite memory n_m. The problem is then formulated in terms of the pdf of the stationary input, denoted by $p(r_{1:n_m})$, where $n_m \leq n_{\text{seq}}$. Assuming that the input signal belongs to a finite set of values $\mathcal{C}$, we can use elements from graph theory to describe any $p(r_{1:n_m})$ in the set of stationary processes as a convex combination of the extreme points of the set. The extreme points are computed as the prime cycles associated with the de Bruijn Graph of memory n_m and alphabet $\mathcal{C}$ (cf. Chapter 2). The proposed input design method is described in page 43, and it will be used in this chapter. The only modification required for the method described in page 43 to cover the contents in this chapter is in step 5, where we will consider the optimization problem described in the next section.

6.2 Input design for identification of closed-loop systems

Problem Definition

Assume that the system (6.2) is controlled using a general (either linear or nonlinear) output feedback controller:

$$u_t = r_t + K_{y,t}(z_{1:t}), \tag{6.15}$$

where $K_{y,t}$ is a known time varying function, r_t denotes an external excitation, and $z_{1:t} := \{y_k - y_d\}_{k=1}^{t}$. The feedback (6.15) is such that the output signal tracks a desired value y_d. To simplify the problem, we assume that y_d is a prescribed constant value. However, the present formulation can include time varying $y_{t,d}$, provided that $\{y_{k,d}\}_{k\geq 1}$ is a stationary process.

The closed-loop structure is shown in Figure 6.1. Thus the closed-loop system

will be

$$\begin{aligned} x_{t+1} &= F_t(\theta, x_t, z_{1:t}) + B(\theta) r_t, \\ y_t &= C(\theta) x_t + \nu_t, \end{aligned} \tag{6.16}$$

where $\nu_t = H(q;\, \theta) e_t$, and

$$F_t(\theta, x_t, z_{1:t}) := A(\theta) x_t + B(\theta) K_{y,t}(z_{1:t}). \tag{6.17}$$

We assume that the resulting closed-loop system (6.16) is asymptotically stable.

Remark 6.1 *If* (6.16) *is asymptotically stable, and the feedback controller is linear, then the information regarding closed-loop stability can be included in the problem formulation by using the Youla parameterization. Since we know that the closed-loop is stable, then we can parameterize the model structure for the system in the set of linear time invariant models that can be stabilized by the given controller, which is performed by using the Youla parameter [90].*

The objective is to design an experiment for the closed-loop system (6.16), that generates n_{seq} samples of the reference signal r_t, to be used for identification of the unknown parameters θ in (6.2). To this end, we consider the experiment design problem (6.14). Since the system is in closed-loop we need to keep the output of the plant y_t close to y_d during the identification experiment. Hence, we choose to minimize the following experimental cost in the optimal input design problem (6.14)

$$J = \mathbf{E} \left\{ \sum_{t=1}^{n_{\text{seq}}} \|y_t - y_d\|_Q^2 + \|\Delta u_t\|_R^2 \right\}, \tag{6.18}$$

where

$$\Delta u_t := u_t - u_{t-1}, \tag{6.19}$$

and Q and R are positive definite matrices. The first term in (6.18) penalizes the deviations from the desired output, while the second term is responsible for minimizing the input energy. The expected value in equation (6.18) is with respect to $\{r_t\}$ and $\{e_t\}$.

In practical applications, it is common to have bounds on the maximal input and output amplitudes allowed by the process. These constraints appear due to physical limitations and/or to keep the system in a safe operating point. However, these bounds cannot be prescribed in a deterministic sense, since both the input and output of the system contains a stochastic process that cannot be measured by the user. Therefore, it is necessary to quantify the input and output constraints by using a probability measure. Thus, we consider the following probabilistic constraints during the identification process[1]:

$$\begin{aligned} &\mathbf{P}\{|y_t - y_d| \leq y_{\max}\} > 1 - \epsilon_y, \;\; t = 1, \ldots, n_{\text{seq}}, \\ &\mathbf{P}\{|u_t| \leq u_{\max}\} > 1 - \epsilon_x, \;\; t = 1, \ldots, n_{\text{seq}}, \end{aligned} \tag{6.20}$$

[1] In equation (6.20) we consider absolute value and the inequality as element-wise operations.

where $u_{\max}$ is the maximum allowed value for the input signal and $y_{\max}$ is the maximum allowed deviation of the output from its desired value, based on the physical properties of the system and actuators; and ϵ_x, $\epsilon_y \geq 0$ are two design variables that define the desired probabilities of being in the safe bounds for input and output signals.

In addition to the previous constraints, we require that the updated (or newly) designed controller based on the estimated parameters can guarantee an acceptable control performance, i.e., the experiment design constraint (6.5) is satisfied. The optimization problem can be summarized as:

Problem 6.1 *Design* $\{r_t^{\text{opt}}\}_{t=1}^{n_{\text{seq}}}$ *as the solution of*

$$
\begin{aligned}
\min_{\{r_t\}_{t=1}^{n_{\text{seq}}}} \quad & J = \mathbf{E}\left\{\sum_{t=1}^{n_{\text{seq}}} \|y_t - y_d\|_Q^2 + \|\Delta u_t\|_R^2\right\}, \\
\text{s. t.} \quad & x_{t+1} = F_t(\theta, x_t, z_{1:t}) + B(\theta) r_t, \\
& y_t = C(\theta) x_t + \nu_t, \ t = 1, \ldots, n_{\text{seq}}, \\
& \nu_t = H(q;\theta) e_t, \ t = 1, \ldots, n_{\text{seq}}, \\
& u_t = r_t + K_{y,t}(z_{1:t}), \quad t = 1, \ldots, n_{\text{seq}}, \\
& z_t = y_t - y_d, \quad t = 1, \ldots, n_{\text{seq}}, \\
& \mathbf{P}\{|y_t - y_d| \leq y_{\max}\} > 1 - \epsilon_y, \ t = 1, \ldots, n_{\text{seq}}, \\
& \mathbf{P}\{|u_t| \leq u_{\max}\} > 1 - \epsilon_x, \ t = 0, \ldots, n_{\text{seq}} - 1, \\
& \mathcal{I}_F^e \succeq \frac{\gamma \chi_\alpha^2(n)}{2} \nabla_\theta^2 V_{\text{app}}(\theta^\star)\big|_{\theta^\star = \theta},
\end{aligned}
\tag{6.21}
$$

where $\mathcal{I}_F$ *is the Fisher information matrix obtained with* n_{seq} *samples.*

Note that Problem 6.1 has a very similar structure to model predictive control (MPC), see [64]. However, they are not necessarily the same since we are not considering a receding horizon approach in this problem.

The optimization problem (6.21) is nonconvex due to the possible nonlinearity of the closed-loop system and the experiment design constraints, and is difficult to solve explicitly.

Convex relaxation of the optimization algorithm

To find a convex relaxation of Problem 6.1 we will use elements from graph theory discussed in Chapters 3 and 4 to design an input sequence $r_{1:n_{\text{seq}}}$ such that the cost function in (6.21) is minimized, satisfying input-output requirements, and identification constraints. To proceed we will assume that $r_{1:n_{\text{seq}}}$ is a realization from a stationary pdf $p(r_{1:n_m})$, where $n_m \leq n_{\text{seq}}$ is the memory of the stationary process. In addition, we assume that $r_t \in \mathcal{C}$, for $t \in \{1, \ldots, n_{\text{seq}}\}$, with $\mathcal{C}$ defined as a set with finite cardinality $n_{\mathcal{C}}$. Under the previous assumptions, we can use the

methods introduced in Chapters 3 and 4 to define the set of feasible pdfs $p(r_{1:n_m})$ as a convex combination of its extreme points. The extreme points are computed as the prime cycles associated with the de Bruijn graph of memory n_m and alphabet $\mathcal{C}$ (cf. Theorem 2.1 in page 28).

If the set of probability mass functions associated with the extreme points is given by $\{p_j\}_{j=1}^{n_v}$, then the optimization problem can be rewritten as

$$
\begin{aligned}
\min_{\{\beta_1,\ldots,\beta_{n_v}\}} \quad & J = \sum_{t=1}^{n_{\text{seq}}} \sum_{j=1}^{n_v} \beta_j \mathbf{E}_{e_t, r_t^{(j)}} \left\{ \left\| y_t^{(j)} - y^d \right\|_Q^2 + \left\| \Delta u_t^{(j)} \right\|_R^2 \right\}, \\
\text{s.t.} \quad & x_{t+1}^{(j)} = F_t(\theta, x_t^{(j)}, z_{1:t}^{(j)}) + B(\theta) r_t^{(j)}, \\
& y_t^{(j)} = C(\theta) x_t^{(j)} + \nu_t, \ t = 1, \ldots, n_{\text{seq}}, \\
& \nu_t = H(q;\, \theta) e_t, \ t = 1, \ldots, n_{\text{seq}}, \\
& u_t^{(j)} = r_t^{(j)} - K_{y,t}(z_{1:t}^{(j)}), \quad t = 1, \ldots, n_{\text{seq}}, \\
& z_t^{(j)} = y_t^{(j)} - y_d, \quad t = 1, \ldots, n_{\text{seq}}, \\
& \sum_{j=1}^{n_v} \beta_j \mathbf{P}_{e_t, r_t^{(j)}} \{ |u_t^{(j)}| \leq u_{\max} \} > 1 - \epsilon_x, \\
& \sum_{j=1}^{n_v} \beta_j \mathbf{P}_{e_t, r_t^{(j)}} \{ |y_t^{(j)} - y_d| \leq y_{\max} \} > 1 - \epsilon_y, \\
& \sum_{j=1}^{n_v} \beta_j \, \mathcal{I}_F^{(j)} \succeq \frac{\gamma \chi_\alpha^2(n)}{2 n_{\text{seq}}} \, \nabla_\theta^2 V_{\text{app}}(\theta^\star) \big|_{\theta^\star = \theta}, \\
& \sum_{j=1}^{n_v} \beta_j = 1, \ \beta_j \geq 0, \quad j = 1, \ldots, n_v.
\end{aligned} \tag{6.22}
$$

In the minimization problem (6.22), $r_{1:n_{\text{seq}}}^{(j)}$, $u_{1:n_{\text{seq}}}^{(j)}$, $x_{1:n_{\text{seq}}}^{(j)}$, $y_{1:n_{\text{seq}}}^{(j)}$, and $z_{1:n_{\text{seq}}}^{(j)}$ denote the signals obtained when $r_{1:n_{\text{seq}}}^{(j)}$ is drawn from p_j, $j \in \{1, \ldots, n_v\}$. The objective of the optimization problem (6.22) is to minimize the cost J by finding the normalized weights $\{\beta_j\}_{j=1}^{n_v}$ satisfying the probabilistic bounds in the input and output, and the requirement on the accuracy of the parameter estimates. The first five equations in the constraints of (6.22) are the equations given by the closed loop system, which are defined for the external excitations $\{r_{1:n_{\text{seq}}}^{(j)}\}_{j=1}^{n_v}$. The following equations represent the probabilistic bounds for the input and the output, the constraint on the accuracy of the parameter estimates, and the requirement that $\{\beta_j\}_{j=1}^{n_v}$ are nonnegative, normalized weights. We notice that the probabilistic bounds, and the requirement on the accuracy of the parameter estimates are expressed as convex combination of the respective quantities achieved by the external excitations $\{r_{1:n_{\text{seq}}}^{(j)}\}_{j=1}^{n_v}$.

If we denote by $\{\beta_j^{\text{opt}}\}_{j=1}^{n_v}$ the set of weighting factors minimizing (6.22), then the optimal pdf is given by

$$p^{\text{opt}} := \sum_{j=1}^{n_v} \beta_j^{\text{opt}} p_j \,. \tag{6.23}$$

Since the set $\{p_j\}_{j=1}^{n_v}$ is known (each p_j is a uniform distribution over the nodes in the j-th prime cycle, cf. Theorem 2.1 in page 28), we can sample $\{r_t^{(j)}\}_{t=1}^{N_{\text{sim}}}$ from each p_j (with N_{sim} sufficiently large), and numerically approximate the expected values $\mathbf{E}_{e_t, r_t^{(j)}}\{\cdot\}$, the probabilities $\mathbf{P}_{e_t, r_t^{(j)}}\{\cdot\}$, and the corresponding information matrices $\mathcal{I}_F^{(j)}$. This approach is based on Chapters 3 and 4, where numerical simulations are employed to compute the information matrices associated with each measure in the set $\{p_j\}_{j=1}^{n_v}$. Indeed, given $\{r_t^{(j)}\}_{t=1}^{N_{\text{sim}}}$ and $\{e_t\}_{t=1}^{N_{\text{sim}}}$, we can generate $\{y_t^{(j)}\}_{t=1}^{N_{\text{sim}}}$ and $\{u_t^{(j)}\}_{t=1}^{N_{\text{sim}}}$ using (6.16). Therefore, we can compute the expressions in (6.22) for each $j \in \{1, \ldots, n_v\}$ as

$$\mathbf{P}_{e_t, r_t^{(j)}}\{|y_t^{(j)} - y_d| \le y_{max}\} \approx \frac{1}{N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \mathbf{1}_{|y_t^{(j)}| \le y_{max}} \,,$$

$$\mathbf{P}_{e_t, r_t^{(j)}}\{|u_t^{(j)}| \le u_{max}\} \approx \frac{1}{N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \mathbf{1}_{|u_t^{(j)}| \le u_{max}} \,,$$

$$\mathbf{E}_{e_t, r_t^{(j)}} \left\{ \left\| y_t^{(j)} - y_d \right\|_Q^2 + \left\| \Delta u_t^{(j)} \right\|_R^2 \right\} \approx \frac{1}{N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \left\{ \left\| y_t^{(j)} - y_d \right\|_Q^2 + \left\| \Delta u_t^{(j)} \right\|_R^2 \right\} ,$$

where $\mathbf{1}_X = 1$ if X is true, and 0 otherwise, and $\Delta u_t^{(j)} = u_t^{(j)} - u_{t-1}^{(j)}$. The computation of $\mathcal{I}_F^{(j)}$ is analyzed in the next subsection.

The key property of the proposed approach is that the input-output constraints and the restriction on the Fisher information matrix are convex in $\{\beta_j\}_{j=1}^{n_v}$. Therefore, the final optimization problem becomes convex in $\{\beta_j\}_{j=1}^{n_v}$.

Computation of the Fisher information matrix

To integrate the experiment design constraint with the optimization problem (6.22), we need to compute the Fisher information matrix $\mathcal{I}_F^{(j)}$ for each $\{r_t^{(j)}\}_{t=1}^{n_{\text{seq}}}$ associated with the j-th extreme point of the set of stationary processes of memory n_m and alphabet $\mathcal{C}$.

We recall that the Fisher information matrix is given by (cf. Equation (4.3a) in Chapter 4)

$$\mathcal{I}_F := \mathbf{E} \left\{ \left. \frac{\partial \log p_\theta(y_{1:n_{\text{seq}}})}{\partial \theta} \right|_{\theta=\theta_0} \left. \frac{\partial \log p_\theta(y_{1:n_{\text{seq}}})}{\partial \theta^\top} \right|_{\theta=\theta_0} \right\} \in \mathbb{R}^{n_\theta \times n_\theta} \,, \tag{6.24}$$

or equivalently,

$$\mathcal{I}_F = -\mathbf{E}\left\{ \frac{\partial^2 \log p_\theta(y_{1:n_{\text{seq}}})}{\partial\theta\partial\theta^\top}\bigg|_{\theta=\theta_0} \right\} \in \mathbb{R}^{n_\theta \times n_\theta} . \tag{6.25}$$

where $y_{1:n_{\text{seq}}} = \{y_t\}_{t=1}^{n_{\text{seq}}}$. In equations (6.24) and (6.25) the expected value is with respect to $\{r_t\}$ and $\{e_t\}$.

Due to the randomness of e_t, (6.16) can be rewritten as

$$\begin{aligned} x_{t+1} &\sim f_\theta(x_{t+1}|y_{1:t}, x_t, r_t) , \\ y_t &\sim g_\theta(y_t|x_t) , \end{aligned} \tag{6.26}$$

where $f_\theta(x_{t+1}|y_{1:t}, x_t, r_t)$ and $g_\theta(y_t|x_t)$ denotes the pdf of the state x_{t+1}, and output y_t, conditioned on the knowledge of $\{x_t,\, r_t\}$ and $y_{1:t}$.

Using the model description (6.26), and the Markov property of the system (6.16), we can write the log-likelihood of the joint distribution of $y_{1:n_{\text{seq}}}$ and $x_{1:n_{\text{seq}}}$ as (cf. Section 4.1)

$$\begin{aligned} \log p_\theta(x_{1:n_{\text{seq}}},\, y_{1:n_{\text{seq}}}) = &\sum_{t=1}^{n_{\text{seq}}-1} \log\{f_\theta(x_{t+1}|y_{1:t}, x_t, r_t)\} \\ &+ \sum_{t=1}^{n_{\text{seq}}} \log\{g_\theta(y_t|x_t)\} + \log\{p_\theta(x_1)\} . \end{aligned} \tag{6.27}$$

To simplify the computations, we will assume that the distribution of the initial state is independent of θ, i.e., $p_\theta(x_1) = p(x_1)$.

Linear feedback

When $K_{y,t}$ is a linear controller, expressions (6.24) and (6.25) can be computed in the frequency domain [59, Section 9.4]. Since we know $\{r_t^{(j)}\}$ for each $j \in \{1, \ldots, n_v\}$, it is possible to compute its corresponding spectrum, say $\Phi_r^{(j)}(\omega)$. To this end, we notice that $\{r_t^{(j)}\}$ is a periodic sequence with period given by T_j. Using [59, Example 2.3] we compute $\Phi_r^{(j)}(\omega)$ as

$$\Phi_r^{(j)}(\omega) = \frac{2\pi}{T_j} \sum_{k=0}^{T_j-1} \Phi_r^{(j),p}(2\pi k/T_j)\delta(\omega - 2\pi k/T_j) ,\ 0 \le \omega < 2\pi , \tag{6.28}$$

where

$$\Phi_r^{(j),\,p}(\omega) := \sum_{\tau=0}^{T_j-1} R_r^{(j)}(\tau) e^{i\omega\tau} , \tag{6.29}$$

$$R_r^{(j)}(\tau) := \frac{1}{T_j} \sum_{t=1}^{T_j} r_t^{(j)} \left(r_{t-\tau}^{(j)}\right)^\top . \tag{6.30}$$

Nonlinear feedback

When K_y is a nonlinear function, equations (6.24) and (6.25) often result in complex (and almost intractable) expressions. Thus, we will approximate the Fisher information matrix using numerical methods. One solution is to use particle methods to approximate (6.24) as the covariance matrix of the gradient of the log-likelihood function, $\nabla_\theta \log p_\theta(y_{1:n_{\text{seq}}})$ (score function) (cf. Chapter 4, Subsection 4.3 [86]). In this case, the score function is estimated using the fixed-lag smoother, and the Fisher information matrix is approximated as the sample covariance matrix over the different realizations of $\nabla_\theta \log p_\theta(y_{1:n_{\text{seq}}})$. Another approach is based on the numerical computation of (6.25) using small perturbation methods [82], where the Hessian (6.25) is computed as an average of numerical approximations based on the score function. The details of the small perturbation method are provided in the next part.

Computing the information matrix using the small perturbation method

Consider the Fisher information matrix given by (6.25). Our goal is to obtain an estimate of (6.25) using the small perturbation method. We use the small perturbation method since it is simple to implement, and that it can approximate (6.25) even if the gradient of the function is not available. In this section we described the technique in [82] to approximate (6.25).

The main idea in [82] is to compute the Hessian (6.25) based on *synthetic data*, i.e., realizations of $\{y_t\}$, and $\{u_t\}$ are obtained by simulation of the model (6.16) for small perturbations of θ_0, using realizations of $\{r_t\}$ and $\{e_t\}$. We will denote the i-th synthetic data as $\mathbf{Z}_{(i)}^{N_{\text{sim}}} = \{y_{(i),t}, u_{(i),t}, r_{(i),t}, e_{(i),t}\}_{t=1}^{N_{\text{sim}}}$, $i \in \{1, \ldots, N\}$. Before we continue, we define $\Delta_{(k,i)} \in \mathbb{R}^{n_\theta}$ as a zero mean random vector such that its entries are independent, identically distributed, symmetrically distributed random variables that are uniformly bounded and satisfy $\mathbf{E}\{\Delta_{(k,i),l}^{-1}\} < \infty$, with $\Delta_{(k,i),l}$ denoting the l-th entry of $\Delta_{(k,i)}$. We notice that the latter condition excludes uniform and Gaussian distributions. Furthermore, we assume that $\Delta_{(k,i),l}$ are bounded in magnitude. A valid choice is the Bernoulli distribution, defined on $\{-1, 1\}$, and with parameter $\eta = 0.5$.

On the other hand, for each realization $\mathbf{Z}_{(i)}^{N_{\text{sim}}}$, we compute the k-th approximation of (6.25) ($k \in \{1, \ldots, M\}$) as

$$\hat{\mathcal{I}}_F^{(k,i)} := -\frac{1}{2}\left\{ \frac{\delta G_{(k,i)}}{2c}\left(\Delta_{(k,i)}^{-1}\right)^\top + \Delta_{(k,i)}^{-1}\frac{\delta G_{(k,i)}^\top}{2c} \right\}, \tag{6.31}$$

where ,

$$\delta G_{(k,i)} := \nabla_\theta \log p_\theta(y_{(i),1:N_{\text{sim}}})\big|_{\theta=\theta_0+c\Delta_{(k,i)}} - \nabla_\theta \log p_\theta(y_{(i),1:N_{\text{sim}}})\big|_{\theta=\theta_0-c\Delta_{(k,i)}} \tag{6.32}$$

$\Delta_{(k,\,i)}^{-1}$ denotes the entry-wise inverse vector of $\Delta_{(k,\,i)}$, $y_{(i),\,1:N_{\mathrm{sim}}} = \{y_{(i),\,t}\}_{t=1}^{N_{\mathrm{sim}}}$, and $c > 0$ is a predefined constant. In [81] is shown that, under suitable assumptions, the bias of the Fisher information matrix estimate (6.31) is of order $\mathcal{O}(c^2)$. The main assumption in [81] is the smoothness of $\log p_\theta(y_{1:N_{\mathrm{sim}}})$, which is reflected in the assumption that $\nabla_\theta \log p_\theta(y_{1:N_{\mathrm{sim}}})$ is thrice continuously differentiable in a neighbourhood of θ_0. However, the main limitation of (6.31) is that it is a low rank approximation of the Fisher information matrix ($\hat{\mathcal{I}}_F^{(k,\,i)}$ has at most rank two), and it may not even be positive semi-definite.

In [82] an improved estimate of the Fisher information matrix is provided as a function of (6.31). To this end, we define $D_{(k,\,i)} \in \mathbb{R}^{n_\theta \times n_\theta}$ as

$$D_{(k,\,i)} := \Delta_{(k,\,i)} \left(\Delta_{(k,\,i)}^{-1}\right)^\top - I\,, \tag{6.33}$$

and the function $\Psi_{(k,\,i)} : \mathbb{R}^{n_\theta \times n_\theta} \to \mathbb{R}^{n_\theta \times n_\theta}$ as

$$\Psi_{(k,\,i)}(H) := \frac{1}{2} H D_{(k,\,i)} + \frac{1}{2} D_{(k,\,i)}^\top H\,. \tag{6.34}$$

Based on (6.33) and (6.34), we can write the new Fisher information matrix estimate in recursive form (in i) as

$$\overline{\mathcal{I}}_F^{(M,\,i)} = \frac{i-1}{i} \overline{\mathcal{I}}_F^{(M,\,i-1)} + \frac{1}{iM} \sum_{k=1}^{M} \left[\hat{\mathcal{I}}_F^{(k,\,i)} + \Psi_{(k,\,i)}\left(\overline{\mathcal{I}}_F^{(M,\,i-1)}\right)\right]\,, \tag{6.35}$$

where $\overline{\mathcal{I}}_F^{(M,\,0)} = 0$. In [82] is shown that the estimate (6.35) tends to $\mathcal{I}_F + \mathcal{O}(c^2)$ as $N \to \infty$ (for a fixed M) in mean square sense. The advantage of using the estimator (6.35) over (6.31) is that (6.35) guarantees that the estimate will be a full rank and positive-definite matrix.

6.3 Numerical example

To illustrate the previous discussion, we introduce the following example:

Example 6.1 *Consider the open-loop, SISO state space system described by*

$$x_{t+1} = \theta_2^0\, x_t + u_t\,, \tag{6.36a}$$
$$y_t = \theta_1^0\, x_t + e_t\,, \tag{6.36b}$$

with true parameters $\theta_0 = \begin{bmatrix}\theta_1^0 & \theta_2^0\end{bmatrix}^\top = \begin{bmatrix}0.6 & 0.9\end{bmatrix}^\top$. *The system is controlled in closed-loop using the controller*

$$u_t = r_t - k_y\, y_t\,, \tag{6.37}$$

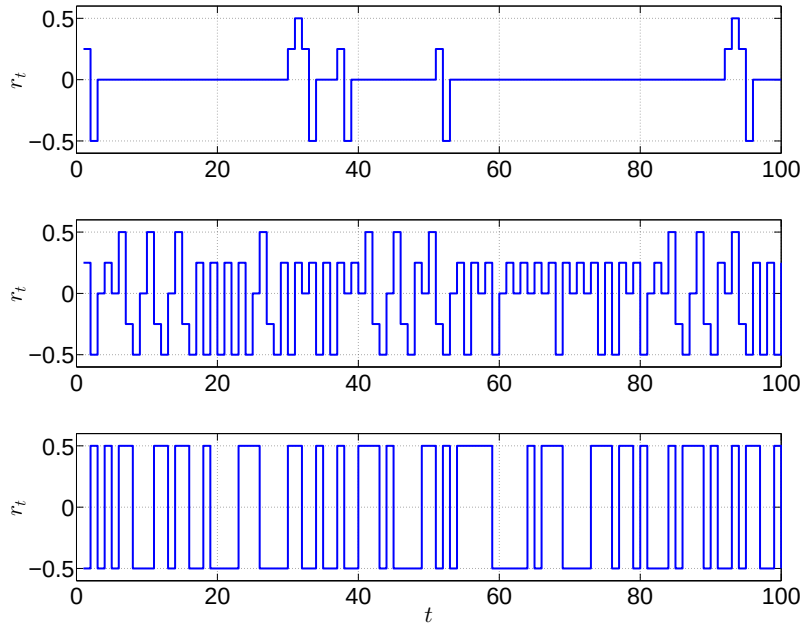

Figure 6.2: Part of the reference signal $\{r_t\}_{t=1}^{500}$. Top: Optimal reference signal. Middle: Optimal reference signal without probabilistic constraints. Bottom: Random binary signal.

where $k_y = 0.5$ *is a known constant. The objective is to identify the open-loop parameters* $\theta = \begin{bmatrix}\theta_1 & \theta_2\end{bmatrix}^\top$ *from the identified closed-loop ones* $\theta_c = \begin{bmatrix}\theta_1^c & \theta_2^c\end{bmatrix}^\top$ *in the model*

$$x_{t+1} = \theta_2^c\, x_t + r_t - k_y\, e_t\,, \tag{6.38a}$$
$$y_t = \theta_1^c\, x_t + e_t\,, \tag{6.38b}$$

using the transformation law

$$\theta_1 = \theta_1^c\,, \tag{6.39a}$$
$$\theta_2 = \theta_1^c + k_y\, \theta_1^c\,. \tag{6.39b}$$

To this end, we will design the reference signal $\{r_t\}_{t=1}^{500}$ *as a realization of a stationary process with memory* $n_m = 2$*, and subject to* $r_t \in \mathcal{C} = \{-0.5,\, -0.25,\, 0,\, 0.25,\, 0.5\}$*, for all* $t \in \{1, \ldots, 500\}$*. Since the experiment will be performed in closed-loop, we*

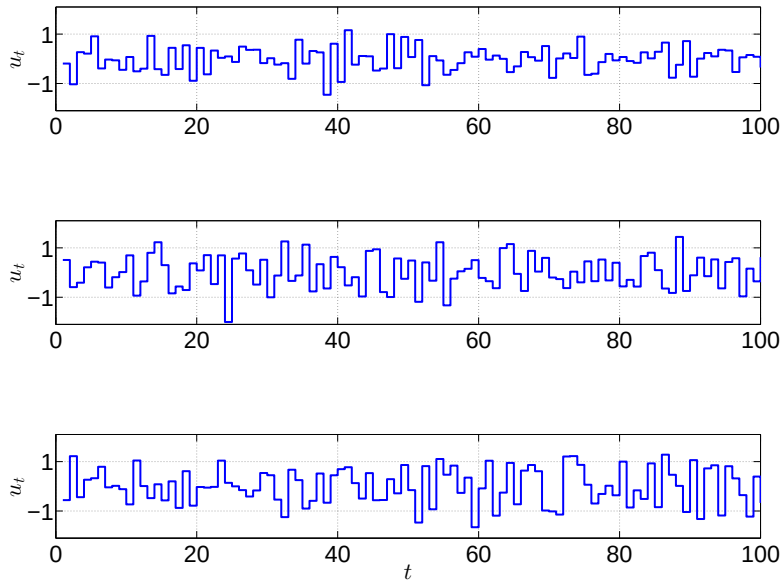

Figure 6.3: Part of the input signal $\{u_t\}_{t=1}^{500}$. Top: Input signal for the optimal reference. Middle: Input signal for the optimal reference without probabilistic constraints. Bottom: Input signal for a random binary reference.

define the following cost function to measure performance degradation:

$$V_{\text{app}}(\theta) = \frac{1}{500}\sum_{t=1}^{500} \|y_t(\theta_0) - y_t(\theta)\|_2^2\,, \tag{6.40}$$

where $y_t(\theta)$ denotes the closed-loop output when θ is employed to describe the open loop model and a linear output feedback controller with constant gain k_y has been used. The cost function (6.40) *is used to build the application set as in* (6.9)*. Finally, we will solve the approximate problem* (6.22)*, where $y_d = 0$, for all $t \in \{1, \ldots, 500\}$, $Q = 1$, $R = 0.02$, $\varepsilon_y = \varepsilon_x = 0.07$, $y_{max} = 2$, $u_{max} = 1$, $\gamma = 10^2$, and $\alpha = 0.98$.*

Figure 6.2 presents one realization of $\{r_t\}_{t=1}^{500}$ obtained by solving (6.22)*, one realization of $\{r_t\}_{t=1}^{500}$ obtained by solving* (6.22) *without probabilistic constraints, and from a random binary sequence with values $\{-0.5, 0.5\}$. From this figure we see that the optimal sequence is zero most of the time, except for short pulses. This can be explained from the tight probabilistic bounds imposed for $\{u_t\}$, which*

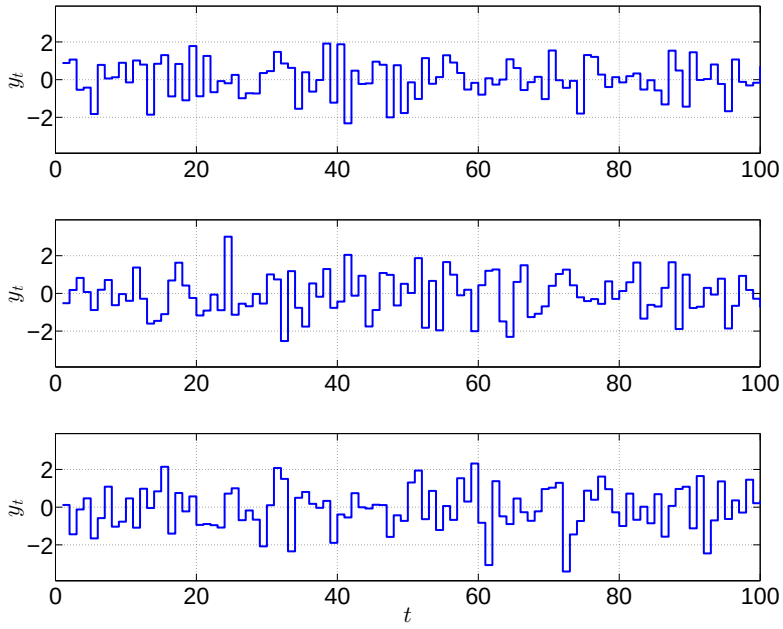

Figure 6.4: Part of the output $\{y_t\}_{t=1}^{500}$. Top: Output signal for the optimal reference. Middle: Output signal for the optimal reference without probabilistic constraints. Bottom: Output signal for a random binary reference.

restricts the excitation provided by $\{r_t\}$*. If we compare the previous signal with the one obtained by solving* (6.22) *without probabilistic bounds, we see that the reference signal contains more oscillations when the probabilistic bounds are removed.*

Figures 6.3 and 6.4 present one realization for the resulting input $\{u_t\}_{t=1}^{500}$ *and output* $\{y_t\}_{t=1}^{500}$*, respectively. From those realizations, we conclude that, for the optimal reference, the input and output are inside the limiting regions* 93.8%*, and* 96% *of the time, respectively, which satisfies the design requirements. On the other hand, for the reference signal obtained by solving* (6.22) *without probabilistic bounds, we have that the input and output satisfies the constraints* 86.6% *and* 93.4% *of the time, respectively. Therefore, in this example we need to incorporate the probabilistic bounds to guarantee that both the input and output of the system are inside the desired region with the prescribed confidence level. With the previous modification, we restrict the set of optimal feasible solutions for the problem of minimum variance to the subset of optimal solutions satisfying the probabilistic bounds. Finally, for the random binary reference, we have that the input and output are inside the confidence*

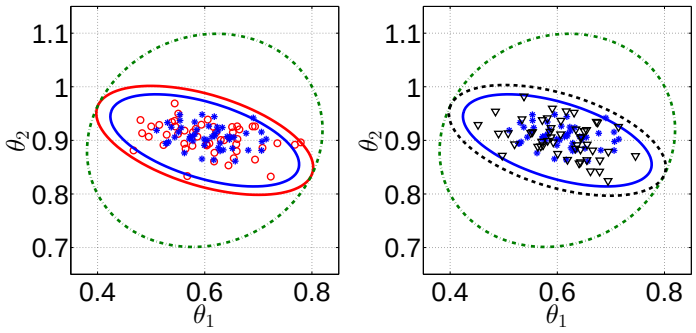

Figure 6.5: Application ellipsoid (green, dot-dashed line) with the respective identification ellipsoids. Blue, continuous line: Identification ellipsoid for the random binary reference (realizations marked with $*$). Red, continuous line: Identification ellipsoid for the optimal reference with probabilistic bounds (realizations marked with circles). Black, dashed line: Identification ellipsoid for the optimal reference without probabilistic bounds (realizations marked with triangles).

region 90.8%*, and* 79.6% *of the time, which does not satisfy the confidence bounds for the system.*

To analyze the identification performance, Figure 6.5 presents the application ellipsoid for the parameter θ*, together with the resulting identification ellipsoids and* 50 *identified parameters obtained with the optimal reference with probabilistic bounds, the optimal reference without probabilistic bounds, and for the random binary reference. From this figure we conclude that the* 98% *confidence level set for the identified parameters lies completely inside the application ellipsoid for all the reference signals. As expected, the confidence level set for the random binary reference is smaller than the ones obtained with the proposed technique, since the variance of this signal is greater than the one obtained with the optimal references. Hence, the random binary reference excites the system more than required, which makes the cost function in optimization problem* (6.21) *greater than the cost obtained with the proposed method. Indeed, the cost functions are* $J^{\text{opt}} = 541.6$ *for the optimal experiment with probabilistic bounds, and* $J^{\text{binary}} = 695.8$ *for a random binary reference, which is in line with the size of the uncertainty ellipsoids in Figure 6.5. On the other hand, we see that the confidence ellipsoids for the estimated parameters are almost the same when a reference signal is designed by including or excluding the probabilistic bounds on the input and output.*

6.4 Conclusion

In this chapter a method to design input sequences for closed-loop experiments has been proposed. The method considers the input sequence as a realization of a stationary process minimizing the experimental cost, and subject to performance constraints. Using the graph-theoretical approach presented in Chapters 2-4, we can express both experimental cost and constraints as a convex combination of the values associated with the extreme measures in the set of stationary processes with finite alphabet and memory, which are computed using numerical methods. An interesting feature of this approach is that probabilistic constraints become convex in the decision variables. The numerical example shows that this approach is an attractive method for the design of input sequences to identify models in a closed-loop setting.

Chapter 7

Conclusions

In this thesis we have introduced a new approach to input design for system identification. The results presented in this thesis can be seen as an extension of the results in [54] and [6, 7]. The main difference with [6, 7] is that we optimize over the stationary probability mass function (pmf) associated with the Markov chain, instead of directly optimizing over the transition probabilities, which results in a convex problem. On the other hand, this thesis covers the optimal input design problem for identification of nonlinear output error and nonlinear state space models, which extends the nonlinear FIR model structure considered in [54].

The proposed technique considers the optimization of an input sequence as a realization of a stationary process with finite memory and finite alphabet, which maximizes a scalar cost function of the Fisher information matrix. By using notions of graph theory, it is possible to characterize the set of stationary probability mass functions (pmfs) through its extreme points. It has been proved that the extreme points of the set of stationary processes are in one to one correspondence with the prime cycles in the equivalent de Bruijn graph.

Once the prime cycles are computed, the probability measures associated with the extreme points of the set of stationary processes are known. Furthermore, an input sequence can be drawn from each prime cycle, which can be employed to compute a numerical approximation of the Fisher information matrix for each extreme point in the set. Therefore, the Fisher information matrix can be computed with a prescribed accuracy even if closed form expressions are not available, which makes the proposed input design method suitable for nonlinear model structures. The Fisher information matrices computed for the extreme points are then used to solve the input design problem by finding the optimal convex combination of the given matrices. Thus, the input design problem becomes convex even for nonlinear model structures.

The solution of the input design problem delivers the optimal stationary pmf associated with the optimal cost function, from which an input sequence must be sampled. The problem of obtaining a sample from a prescribed stationary pmf

can be solved by using Markov chains, where the input samples are obtained as the states of the chain. Provided that the Markov chain is ergodic, the input samples will tend to be distributed according to the stationary distribution of the Markov chain. The existing results on the design of Markov chains with a prescribed stationary distribution lie on the properties of the associated graph and/or the stationary pmf (e.g., undirected graphs, or reversible Markov chains), which are not applicable to the present work. Therefore, this thesis also introduces a method for designing Markov chains for de Bruijn graphs with a prescribed stationary pmf.

As a first approach to input design, we analyze the proposed method for the case of nonlinear output-error model structures. For this case, it is shown that the computation of the Fisher information matrix can be carried out by numerical approximation, where the accuracy of the approximation is controlled directly through the number of samples taken from the associated prime cycle. The numerical examples show that the method retrieves existing results on input design for nonlinear FIR model structures, and that it can also be employed for designing input sequences for identification of more general nonlinear output-error models, which are not covered by previous results in the area.

The major difficulty associated with extending the input design method on Chapter 3 to more general nonlinear models is the approximation of the Fisher information matrices. As a solution to this issue, Chapter 4 introduced an extension of the input design method to nonlinear state space models. The extension relies on particle methods, where an approximation of the observed information matrix is computed as the sample covariance matrix of the score function. In this case, the accuracy of the approximation is a trade-off between the number of particles (which increases the accuracy for a fixed data length) and the data length (which decreases the accuracy for a fixed number of particles). In order to deal with the uncertainty associated with the approximation method, we consider the solution to the optimization problem as the sample mean of the different solutions obtained for different approximations of the Fisher information matrix.

As applications of the proposed method, this thesis discusses two different problems. The first problem concerns the design of input sequence for model estimation in channels with quantized output. Since a numerical approximation for the Fisher information matrix associated with the model is available, the problem can be solved by using the presented method. A simulation example shows that the method can be employed to obtain more informative experiments compared to the results obtained by using a realization from a white noise process.

The second application of the method is found on application oriented closed-loop input design. In this case the objective is the design of an external input sequence to improve the parameters employed for the model of a plant operating in closed-loop, while guaranteeing limits on the performance degradation during the experiment (e.g., probabilistic constraints on the input and output, and confidence ellipsoids for the identified parameters). By using the proposed input design method, the problem can be solved by using convex optimization tools, where both the constraints and the cost function are optimized as a convex combination of the

extreme points characterizing the set of stationary pmfs.

The discussion in this thesis shows that the proposed input design method is an attractive alternative to design input sequences for identification of nonlinear model structures, where the method guarantees a convex optimization problem, which allows to employ powerful tools for solving the optimal input design problem.

Future work

This thesis can be seen as a first approach to input design for identification of nonlinear models. However, there is plenty of space to improve the results in this area. Some ideas for future work in the subject are:

Reduction of computational effort The main limitation of the input design method introduced in this thesis is the computational effort needed to obtain the solution. As discussed in Chapter 2, the computational effort is mostly related with the prime cycles. An approach to overcome this issue is an open question and topic of future research.

Irreducibility of the designed Markov chains The generation of an input realization from a given stationary pmf relies on the fact that the resulting Markov chain is irreducible, i.e., there is only one class of states in the chain. The results presented in this thesis do not guarantee that the optimal pmf will be associated with an irreducible Markov chain, which is important since an input realization distributed according to the desired pmf is needed in order to perform the experiment. A possibility to overcome this issue is to force in the optimization problem the requirement of irreducibility of the Markov chain. Another option for this issue is to reduce the memory of the Markov chain until an irreducible chain is obtained. These options will be analyzed in future research.

Stationary pdfs with continuous support The thesis has focused on the design of input sequences with finite alphabet. An approach to input design when the alphabet lies on a continuous support is an open problem and possible research topic.

Robust optimal input design As with most optimal input design methods, the one presented in this thesis considers the existence of prior knowledge about the model parameters. An alternative to overcome this issue is to implement a robust design scheme on top of it, or through an adaptive input design procedure, where the signal is redesigned as more information is collected from the system.

Application oriented closed-loop experiment design The results obtained in Chapter 6 consider a linear model structure for the plant, under a known nonlinear feedback. An extension of the method to include implicit feedback laws (e.g., as in model predictive control) is a future research direction in the topic.

Appendices

Appendix A

Algorithms for computation of elementary cycles

In this appendix we present the algorithms implemented in the input design methods of Chapters 2-4.

A.1 Preliminaries

Before introducing the algorithms required in Chapters 2-4, we need the following definitions [84]:

Definition A.1 *(Undirected graph) Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph. Then $\mathcal{G}$ is called an* undirected graph *if and only if*

$$\mathcal{E} = \{\, \{v,\, w\} \,|\, v,\, w \in \mathcal{V} \} \;, \tag{A.1}$$

where $\{v,\, w\}$ is an unordered pair of vertices.

Definition A.2 *(Tail and head of an edge) Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed graph. Given an edge $(v,\, w) \in \mathcal{E}$, v is defined as the* tail *of the edge $(v,\, w)$, and w is defined as the* head *of the edge $(v,\, w)$.*

Definition A.3 *(Undirected version of a directed graph) Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed graph. The* undirected version of a directed graph *$\mathcal{G}$ is defined as the graph formed by converting each edge in $\mathcal{E}$ into an undirected edge, and removing duplicate edges.*

Definition A.4 *(Connected undirected graph) Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a undirected graph. Then $\mathcal{G}$ is said to be* connected *if there is a path between every pair of vertices.*

Definition A.5 *(Tree) A* tree *$\mathcal{T}$ is a directed graph whose undirected version is connected. In addition, a tree $\mathcal{T}$ has one vertex which is the head of no edges (called*

the root*), and such that all vertices except the root are head of exactly one edge. If* (v, w) *is an edge of* $\mathcal{T}$*, then* v *is an* ancestor *of* w*, and* w *is a* descendant *of* v*.*

Definition A.6 *(Subtree) Let* $\mathcal{T}$ *be a tree, and* v *a node in* $\mathcal{T}$*. Then* $\mathcal{T}_v$ *is the* subtree *of* $\mathcal{T}$ *with respect to* v *if and only if* $\mathcal{T}_v$ *has as vertices all the descendants of* v *in* $\mathcal{T}$*.*

Definition A.7 *(Spanning tree) Let* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *be a directed graph. A tree* $\mathcal{T}$ *is a* spanning tree *of* $\mathcal{G}$ *if* $\mathcal{T}$ *is a subgraph of* $\mathcal{G}$*, and* $\mathcal{T}$ *contains all the vertices of* $\mathcal{G}$*.*

Definition A.8 *(Palm tree) Let* $\mathcal{G}$ *be a directed graph, consisting of two disjoint sets of edges, denoted by* $\mathcal{E}_1$ *and* $\mathcal{E}_2$*, respectively. Suppose* $\mathcal{G}$ *satisfies the following properties:*

(i) The subgraph $\mathcal{T}$ *containing the edges* $\mathcal{E}_1$ *is a spanning tree of* $\mathcal{G}$*.*

(ii) Each edge which is not in the spanning tree $\mathcal{T}$ *of* $\mathcal{P}$ *connects a vertex with one of its ancestors in* $\mathcal{T}$*.*

Then $\mathcal{G}$ *is called a* palm tree*. The edges* $\mathcal{E}_2$ *are called the* fronds *of* $\mathcal{G}$*.*

Definition A.9 *(Cross-link) Let* $\mathcal{T}$ *be a tree, and assume there are subtrees* $\mathcal{T}_v$*,* $\mathcal{T}_w$ *in* $\mathcal{T}$*. Then, the edge* (t_v, t_w) *is called a* cross-link *if and only if* t_v *is in* $\mathcal{T}_v$ *and* t_w *is in* $\mathcal{T}_w$*.*

A.2 Strong connected components of a graph

A term required in the discussion of the algorithms in this thesis is the concept of strongly connected graphs, whose definition is given below [84, Definition 4]:

Definition A.10 *(Strongly connected graph) Let* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *be a directed graph. Suppose that for each pair of vertices* $v, w \in \mathcal{G}$*, there exist paths* $p_1 = (v, \ldots, w)$ *and* $p_2 = (w, \ldots, v)$*. Then* $\mathcal{G}$ *is said to be* strongly connected*.*

Based on Definition A.10, we have the following result [84, Lemma 9]:

Lemma A.1 *(Strongly connected components) Let* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *be a directed graph. We may define an equivalence relation on the set of vertices as follows: two vertices* v *and* w *are equivalent if there is a cycle* $p = (v, \ldots, v)$ *which contains* w*. Let the distinct equivalence classes under this relation be* $\mathcal{V}_i$*,* $i \in \{1, \ldots, n\}$*. Let* $\mathcal{G} = (\mathcal{V}_i, \mathcal{E}_i)$*, where* $\mathcal{E}_i := \{(u, w) \in \mathcal{E} : v, w \in \mathcal{V}_i\}$*. Then:*

(i) Each $\mathcal{G}_i$ *is strongly connected.*

(ii) No $\mathcal{G}_i$ *is a proper subgraph of a strongly connected subgraph of* $\mathcal{G}$*.*

Then the subgraphs $\{\mathcal{G}_i\}_{i=1}^{n}$ *are called the* strongly connected components *of* $\mathcal{G}$*.*

It is shown in [84] that the problem of finding the strongly connected components of a graph $\mathcal{G}$ can be reduced to the problem of finding the roots of the strongly connected components. In this appendix we define LOWLINK(v) as the smallest vertex (according to a user defined indexing for the vertices in $\mathcal{G}$) which is in the same component as v and is reachable by traversing zero or more tree arcs followed by at most one frond or cross-link. We refer to [84, p. 156] for more details about the definition of this function.

Algorithm A.1 in page 118 presents a pseudo-code for computing the set of strongly connected components in a given graph $\mathcal{G}$ [84, p. 157]. It is shown in [84] that Algorithm A.1 requires $\mathcal{O}(\mathcal{V}, \mathcal{E})$ space and time, where

$$\mathcal{O}(\mathcal{V}, \mathcal{E}) \leq k_1 \, \#\mathcal{V} + k_2 \, \#\mathcal{E} + k_3 \,, \tag{A.2}$$

for some positive constants k_1, k_2, and k_3.

The pseudo-code for determining the set of strongly connected components in a graph $\mathcal{G}$ will be employed in the computation of elementary cycles in $\mathcal{G}$, which is discussed in the next section.

A.3 Elementary cycles of a graph

As it is mentioned in Chapter 2, the computation of prime cycles in a graph $\mathcal{G}_{\mathcal{C}^{n_m}}$ can be performed by finding all the elementary cycles in $\mathcal{G}_{\mathcal{C}^{n_m-1}}$. In this section we provide an algorithm to find the set of elementary cycles in a directed graph $\mathcal{G}$. The algorithm is based on the one introduced in [46, pp. 79–80]. The pseudo-code associated with this algorithm is presented in Algorithm A.2-A.3.

We describe Algorithm A.2-A.3 based on the discussion provided in [46, p. 79]. The algorithm proceeds by building elementary paths from s. The vertices of the current elementary path are kept on a stack. A vertex is appended to an elementary path by a call to the procedure CIRCUIT and is deleted upon return from this call. When a vertex v is appended to a path it is blocked by setting BLOCKED(v) = true, so that v cannot be used twice on the same path. However, when we return from the call which blocks v, v is not necessarily unblocked. The idea is that we unblock a node with a sufficient delay such that any two unblockings of v are separated by either an output of a new circuit or a return to the main procedure.

Algorithm A.1 Computation of strongly connected components in a graph

INPUTS: A directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
OUTPUT: The set of strongly connected components in $\mathcal{G}$.

```
1: Begin
2: Integer i;
3: Procedure STRONGCONNECT(v);
4:     Begin
5:     i := i + 1;
6:     NUMBER(v) := i;
7:     LOWLINK(v) := NUMBER(v);
8:     put v on stack of points;
9:     For w in the set of descendants D_v do
10:        Begin
11:        If w is not yet numbered then
12:            Begin comment (v, w) is a tree arc;
13:            STRONGCONNECT(w);
14:            LOWLINK(v) := min{LOWLINK(v), NUMBER(w)};
15:            end
16:        Else If NUMBER(w) < NUMBER(v) do
17:            Begin comment (v, w) is a frond or cross-link;
18:            If w is on stack of points then
19:                Begin
20:                LOWLINK(v) := min{LOWLINK(v), NUMBER(w)};
21:                end
22:            end
23:        end
24:    If LOWLINK(v) = NUMBER(v) then
25:        Begin comment v is the root of a component;
26:        start new strongly connected component;
27:        While w on top of point stack satisfies NUMBER(w) ≥ NUMBER(v) do
28:            Begin
29:            delete w from point stack and put w in current component;
30:            end
31:        end
32:    end
33: i := 0;
34: empty stack of points;
35: For w a vertex do
36:    Begin
37:    If w is not yet numbered then
38:        Begin
39:        STRONGCONNECT(w);
40:        end
41:    end
42: end
```

Algorithm A.2 Computation of elementary cycles in a graph (Part I)

INPUTS: A directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
OUTPUT: The set of elementary cycles in $\mathcal{G}$.

```
 1: Begin
 2: Integer list array A_K(n), B(n);
 3: Logical array BLOCKED(N);
 4: Integer s;
 5: Logical Procedure CIRCUIT(v);
 6:     Begin Logical f;
 7:     Procedure UNBLOCK(u);
 8:         Begin
 9:         BLOCKED(u) := false;
10:         For w ∈ B(u) do
11:             Begin
12:             delete w from B(u);
13:             If BLOCKED(w) then
14:                 Begin
15:                 UNBLOCK(w);
16:                 end
17:             end
18:         end
19:     f := false;
20:     stack v;
21:     BLOCKED(v) := true;
22:     For w ∈ A_K(v) do
23:         Begin
24:         If w = s then
25:             Begin
26:             output circuit composed of stack followed by s;
27:             f := true;
28:             end
29:         Else If BLOCKED(w) = false then
30:             Begin
31:             If CIRCUIT(w) = true then
32:                 Begin
33:                 f := true;
34:                 end
35:             end
36:         end
37:     If f = true then
38:         Begin
39:         UNBLOCK(v);
40:         end
41:     Else For w ∈ A_K(v) do
42:         Begin
43:         If v ∉ B(w) then
44:             Begin
45:             put v on B(w);
46:             end
47:         end
48:     unstack v;
49:     CIRCUIT := f;
50:     end
```

Algorithm A.3 Computation of elementary cycles in a graph (Part II)

INPUTS: A directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
OUTPUT: The set of elementary cycles in $\mathcal{G}$.

```
51: empty stack;
52: s := 1;
53: while s < n do
54:     Begin
55:     A_K := adjacency matrix of strong component K with least vertex in subgraph
56:             of G induced by {s, s + 1, ..., n};
57:     If A_K ≠ ∅ then
58:         Begin
59:         s := least vertex in V_K;
60:         For i ∈ V_K do
61:             Begin
62:             BLOCKED(i) := false;
63:             B(i) := ∅;
64:             end
65:         dummy := CIRCUIT(s);
66:         s := s + 1;
67:         end
68:     Else
69:         Begin
70:         s := n;
71:         end
72:     end
73: end
```

Appendix B

Convergence of the approximation of $\mathcal{I}_F$

In this appendix we prove that the approximation (3.17) in page 42 converges to $\mathcal{I}_F^{(i)}$ as $N_{\text{sim}} \to \infty$:

Theorem B.1 *If $\{u_t\}$ has period T satisfying $|u_t| \leq K$ for some $K \geq 0$, and $\psi_t^{\theta_0}(u_t)\psi_t^{\theta_0}(u_t)^\top$ is exponentially stable, then*

$$\begin{aligned}\lim_{N_{\text{sim}}\to\infty} \frac{1}{N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top \\ = \frac{1}{T}\sum_{t=1}^{T} \psi_t^{\theta_0}(u_t)\psi_t^{\theta_0}(u_t)^\top \\ = \int \psi_t^{\theta_0}(u_{t,-\infty})\psi_t^{\theta_0}(u_{t,-\infty})^\top \, dP(u_{t,-\infty})\,,\end{aligned} \tag{B.1}$$

where $\{\tilde{u}_t\}$ is equal to $\{u_t\}$ for $t > 0$ but $\tilde{u}_t = 0$ for $t \leq 0$, $\tilde{u}_{t,-\infty} := \{\tilde{u}_k\}_{k=-\infty}^{t}$, $u_{t,-\infty} := \{u_k\}_{k=-\infty}^{t}$, and P is the probability measure of a Markov chain generating $\{u_t\}$ (a uniform probability distribution on the set of possible values of $u_{1:T}$).

Proof *Given $\varepsilon > 0$, take S as a multiple of T such that $C\delta_\psi^S < \varepsilon$. Then, for every $t > S$,*

$$\left|\psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top - \psi_t^{\theta_0}(\tilde{u}_t^{t-S})\psi_t^{\theta_0}(\tilde{u}_t^{t-S})^\top\right| < C\delta_\psi^S < \varepsilon\,. \tag{B.2}$$

On the other hand, since $\{u_t\}$ is periodic of period T, $u_{t,-\infty}^{t-S}$ takes only a finite number of values for $t > S$ (at most S), we have that for $N_{\text{sim}} = mS + n$ (with m, n positive integers $|n| \leq S$):

$$\frac{1}{N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top = \frac{mS}{N_{\text{sim}}} \frac{1}{mS} \left[\sum_{t=1}^{mS} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top\right.$$

$$+ \sum_{t=mS+1}^{N_{\text{sim}}} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top \Bigg]$$

$$= \frac{mS}{N_{\text{sim}}} \frac{1}{mS} \sum_{t=1}^{mS} \left[\psi_t^{\theta_0}(\tilde{u}_t^{t-S})\psi_t^{\theta_0}(\tilde{u}_t^{t-S})^\top + \eta_t\right] + \frac{1}{N_{\text{sim}}} \sum_{t=mS+1}^{N_{\text{sim}}} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top$$

$$= \frac{mS}{N_{\text{sim}}} \frac{1}{T} \sum_{t=1}^{T} \psi_t^{\theta_0}(u_t^{t-S})\psi_t^{\theta_0}(u_t^{t-S})^\top + \frac{1}{N_{\text{sim}}} \sum_{t=1}^{mS} [\mu_t + \eta_t]$$

$$+ \frac{1}{N_{\text{sim}}} \sum_{t=mS+1}^{N_{\text{sim}}} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top , \quad \text{(B.3)}$$

where μ_t, $\eta_t \in [-\varepsilon, \varepsilon]$. *Thus, the second term in* (B.3) *is bounded by* 2ε. *Moreover, the third term in* (B.3) *tends to* 0 *as* $N_{\text{sim}} \to \infty$ *(since it consists of a sum of a most* S *terms). Therefore,*

$$\left| \lim_{N_{\text{sim}} \to \infty} \frac{1}{N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top - \frac{1}{T} \sum_{t=1}^{T} \psi_t^{\theta_0}(u_t^{t-S})\psi_t^{\theta_0}(u_t^{t-S})^\top \right| \leq 2\varepsilon , \quad \text{(B.4)}$$

and since ε *was arbitrary, we conclude that*

$$\lim_{N_{\text{sim}} \to \infty} \frac{1}{N_{\text{sim}}} \sum_{t=1}^{N_{\text{sim}}} \psi_t^{\theta_0}(\tilde{u}_t)\psi_t^{\theta_0}(\tilde{u}_t)^\top = \frac{1}{T} \sum_{t=1}^{T} \psi_t^{\theta_0}(u_t^{t-S})\psi_t^{\theta_0}(u_t^{t-S})^\top . \quad \text{(B.5)}$$

The last equality in (B.1) *follows since* P *assigns equal probability to* T *different sequences (corresponding to the possible sequences obtained by shifting* $\{u_t\}$*).* ■

Appendix C

The expectation-maximization algorithm

In this appendix we present the expectation-maximization algorithm, and useful identities for the methods discussed in Chapter 4. The material in this appendix is based on [8, 76, 77].

C.1 The expectation-maximization algorithm

The expectation-maximization (EM) algorithm [20, 66] is an iterative procedure that at the k-th step seeks a value θ_k such that the likelihood is increased in the sense that $\log p_{\theta_k}(y_{1:T}) > \log p_{\theta_{k-1}}(y_{1:T})$.

The main idea of the EM algorithm is the postulation of a *missing* data set $x_{1:T}$. In this thesis, the missing data $x_{1:T}$ will be understood as the state sequence in the model structure (1.3) in page 3, but other choices are possible, and it can be considered as a design variable. Thus, we consider the joint log-likelihood function $\log p_\theta(x_{1:T}, y_{1:T})$ with respect to both the observed data $y_{1:T}$ and the missing data $x_{1:T}$. This approach assumes that maximizing the joint log-likelihood $\log p_\theta(x_{1:T}, y_{1:T})$ is easier than maximizing the marginal one $\log p_\theta(y_{1:T})$.

The EM algorithm then copes with $x_{1:T}$ being unavailable by forming an approximation $\mathbf{Q}(\theta, \theta_k)$ of $\log p_\theta(x_{1:T}, y_{1:T})$. The approximation used is the minimum variance estimate of $\log p_\theta(x_{1:T}, y_{1:T})$ given the observed data $y_{1:T}$, and an assumption θ_k of the true parameter value. This minimum variance estimate is given by the conditional expectation [2]:

$$\begin{aligned}\mathbf{Q}(\theta, \theta_k) &:= \mathbf{E}\left\{\log p_\theta(x_{1:T}, y_{1:T}) \middle| y_{1:T}\right\} \\ &= \int_{\mathcal{X}_{1:T}} \log p_\theta(x_{1:T}, y_{1:T}) p_{\theta_k}(x_{1:T}|y_{1:T})\, dx_{1:T}\,. \end{aligned} \tag{C.1}$$

The utility of this approach depends on the relationship between $\log p_\theta(y_{1:T})$, and the approximation $\mathbf{Q}(\theta, \theta_k)$ of $\log p_\theta(x_{1:T}, y_{1:T})$. This can be examined by using

the definition of conditional probability to write

$$\log p_\theta(x_{1:T},\, y_{1:T}) = \log p_\theta(x_{1:T}|y_{1:T}) + \log p_\theta(y_{1:T})\,. \tag{C.2}$$

Taking the conditional expectation $\mathbf{E}\{\cdot|y_{1:T}\}$ on both sides of equation (C.2) we obtain

$$\mathbf{Q}(\theta,\, \theta_k) = \log p_\theta(y_{1:T}) + \int_{\mathcal{X}_{1:T}} \log p_\theta(x_{1:T}|y_{1:T}) p_{\theta_k}(x_{1:T}|y_{1:T})\, dx_{1:T}\,. \tag{C.3}$$

Therefore,

$$\begin{aligned}\log p_\theta(y_{1:T}) - \log p_{\theta_k}(y_{1:T}) &= \mathbf{Q}(\theta,\, \theta_k) - \mathbf{Q}(\theta_k,\, \theta_k) \\ &\quad + \int_{\mathcal{X}_{1:T}} \log \frac{p_{\theta_k}(x_{1:T}|y_{1:T})}{p_\theta(x_{1:T}|y_{1:T})} p_{\theta_k}(x_{1:T}|y_{1:T})\, dx_{1:T}\,. \end{aligned}\tag{C.4}$$

The integral in the right hand side of the equality in (C.4) is known as the *Kullback-Leibler divergence* metric, which is non-negative. Indeed, for $x > 0$ we have $-\log x \geq 1 - x$, which implies

$$\begin{aligned}&-\int_{\mathcal{X}_{1:T}} \log \frac{p_\theta(x_{1:T}|y_{1:T})}{p_{\theta_k}(x_{1:T}|y_{1:T})} p_{\theta_k}(x_{1:T}|y_{1:T})\, dx_{1:T} \\ &\qquad \geq \int_{\mathcal{X}_{1:T}} \left(1 - \frac{p_\theta(x_{1:T}|y_{1:T})}{p_{\theta_k}(x_{1:T}|y_{1:T})}\right) p_{\theta_k}(x_{1:T}|y_{1:T})\, dx_{1:T} = 0\,, \end{aligned}\tag{C.5}$$

where the equality to zero is due to the fact that $p_\theta(x_{1:T}|y_{1:T})$ is of unit area for any value of θ. As a consequence of (C.5) we have that

$$\log p_\theta(y_{1:T}) - \log p_{\theta_k}(y_{1:T}) \geq \mathbf{Q}(\theta,\, \theta_k) - \mathbf{Q}(\theta_k,\, \theta_k)\,. \tag{C.6}$$

Equation (C.6) is the key of the EM algorithm. Namely, choosing θ so that $\mathbf{Q}(\theta,\, \theta_k) > \mathbf{Q}(\theta_k,\, \theta_k)$ implies that the log-likelihood is also increased in that

$$\log p_\theta(y_{1:T}) > \log p_{\theta_k}(y_{1:T}).$$

The EM algorithm exploits this to deliver a sequence of values $\{\theta_k\}$ designed to be increasingly good approximations of the maximum likelihood estimate (1.5) in page 5. Algorithm C.1 summarizes the steps in the EM method.

Remark C.1 *For reference in the next sections, we define*

$$\mathcal{H}(\theta,\, \theta') := -\int_{\mathcal{X}_{1:T}} \log p_\theta(x_{1:T}|y_{1:T}) p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T}\,. \tag{C.8}$$

Algorithm C.1 EM algorithm

INPUTS: $y_{1:T}$ (observations), and $\log p_\theta(x_{1:T}, y_{1:T})$ (joint log-likelihood function).
OUTPUT: $\hat{\theta}_T$ (parameter estimate).

1: Set $k = 0$ and initialize θ_k such that $\log p_\theta(y_{1:T})$ is finite.
2: (Expectation (E) step): Compute $\mathbf{Q}(\theta, \theta_k)$.
3: (Maximization (M) step): Compute

$$\theta_{k+1} = \arg\max_{\theta \in \Theta} \mathbf{Q}(\theta, \theta_k)\,. \tag{C.7}$$

4: If not converged, update $k = k + 1$ and return to line 2. Otherwise, set $\hat{\theta}_T = \theta_{k+1}$, and stop the algorithm.

C.2 EM algorithm: useful identities

In this section we introduce useful results that can be derived from the intermediate expressions for the EM algorithm. In particular, we present the Fisher's and Louis' identities, which are employed to estimate the Fisher information matrix in Chapter 4. To this end, we need the following assumption [8]:

Assumption C.1 *Assume that the following conditions hold:*

(i) The parameter Θ is an open subset of $\mathbb{R}^{n_\theta}$ (for some integer n_θ).

(ii) For any $\theta \in \Theta$, $p_\theta(y_{1:T})$ is positive and finite.

(iii) For any $(\theta, \theta') \in \Theta \times \Theta$,

$$\int_{\mathcal{X}_{1:T}} |\log p_\theta(x_{1:T}|y_{1:T})|\, p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T}\,,$$

is finite.

(iv) $p_\theta(y_{1:T})$ is twice continuously differentiable on Θ.

(v) For any $\theta' \in \Theta$, $\theta \to \mathcal{H}(\theta, \theta')$ is twice continuously differentiable on Θ. In addition,

$$\int_{\mathcal{X}_{1:T}} \left|\nabla_\theta^k \log p_\theta(x_{1:T}|y_{1:T})\right| p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T}\,,$$

is finite for $k = 1, 2$ and any $(\theta, \theta') \in \Theta \times \Theta$, and

$$\nabla_\theta^k \int_{\mathcal{X}_{1:T}} \log p_\theta(x_{1:T}|y_{1:T}) p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T} = \int_{\mathcal{X}_{1:T}} \nabla_\theta^k \log p_\theta(x_{1:T}|y_{1:T}) p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T}\,.$$

Fisher's identity

A result derived from the EM algorithm is known as the *Fisher's identity* [8, 25]:

Theorem C.1 *Consider that Assumption C.1 holds. then:*

$$\nabla_\theta \log p_\theta(y_{1:T})|_{\theta=\theta'} = \int_{\mathcal{X}_{1:T}} \nabla_\theta \log p_\theta(x_{1:T}, y_{1:T})|_{\theta=\theta'}\, p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T}\,, \quad \text{(C.9)}$$

and

$$\begin{aligned} -\nabla^2_\theta \log p_\theta(y_{1:T})\big|_{\theta=\theta'} = \\ -\int_{\mathcal{X}_{1:T}} \nabla^2_\theta \log p_\theta(x_{1:T}, y_{1:T})\big|_{\theta=\theta'}\, p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T} \\ +\int_{\mathcal{X}_{1:T}} \nabla^2_\theta \log p_\theta(x_{1:T}|y_{1:T})\big|_{\theta=\theta'}\, p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T}\,. \end{aligned} \quad \text{(C.10)}$$

Proof *Expression* (C.9) *is* (C.1) *differentiated once under the integral sign (using* (C.3)*), and expression* (C.10) *is* (C.3) *differentiated twice under the integral sign.* ■

Remark C.2 *Expression* (C.9) *is known as* Fisher's identity*, and equation* (C.10) *is normally referred to as the* missing information principle *[62].*

Louis' identity

The second result useful in this thesis is known as *Louis' identity* [8, 62]:

Theorem C.2 *(Louis' identity) Consider that Assumption C.1 holds. Then:*

$$\begin{aligned} \nabla^2_\theta \log p_\theta(y_{1:T})\big|_{\theta=\theta'} + \{\nabla_\theta \log p_\theta(y_{1:T})|_{\theta=\theta'}\}\{\nabla_\theta \log p_\theta(y_{1:T})|_{\theta=\theta'}\}^\top = \\ \int_{\mathcal{X}_{1:T}} \big[\nabla^2_\theta \log p_\theta(x_{1:T}, y_{1:T})\big|_{\theta=\theta'} \\ +\{\nabla_\theta \log p_\theta(x_{1:T}, y_{1:T})|_{\theta=\theta'}\}\{\nabla_\theta \log p_\theta(x_{1:T}, y_{1:T})|_{\theta=\theta'}\}^\top\big]\, p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T}\,. \end{aligned} \quad \text{(C.11)}$$

Proof *To prove* (C.11)*, we start from* (C.10) *and note that the second term on the right-hand side of the equality is the negative of an information matrix for the parameter θ associated with the probability density function $p_\theta(\cdot|y_{1:T})$ evaluated at $\theta = \theta'$. Therefore, we can use the information matrix identity*

$$\int_{\mathcal{X}_{1:T}} \nabla^2_\theta \log p_\theta(x_{1:T}|y_{1:T})\big|_{\theta=\theta'}\, p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T} =$$

$$- \int_{\mathcal{X}_{1:T}} \{ \nabla_\theta \log p_\theta(x_{1:T}|y_{1:T})|_{\theta=\theta'} \} \{ \nabla_\theta \log p_\theta(x_{1:T}|y_{1:T})|_{\theta=\theta'} \}^\top p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T} \,. \quad \text{(C.12)}$$

This is a consequence of Assumption C.1, point (v), and the fact that $p_\theta(\cdot|y_{1:T})$ is a probability density function for all values of θ, implying that

$$\int_{\mathcal{X}_{1:T}} \nabla_\theta \log p_\theta(x_{1:T}|y_{1:T})|_{\theta=\theta'}\, p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T} = 0 \,. \quad \text{(C.13)}$$

Using the identity (C.2)*, and* (C.9) *we conclude that*

$$\begin{aligned} &\int_{\mathcal{X}_{1:T}} \{ \nabla_\theta \log p_\theta(x_{1:T}|y_{1:T})|_{\theta=\theta'} \} \{ \nabla_\theta \log p_\theta(x_{1:T}|y_{1:T})|_{\theta=\theta'} \}^\top p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T} = \\ &\int_{\mathcal{X}_{1:T}} \{ \nabla_\theta \log p_\theta(x_{1:T}, y_{1:T})|_{\theta=\theta'} \} \{ \nabla_\theta \log p_\theta(x_{1:T}, y_{1:T})|_{\theta=\theta'} \}^\top p_{\theta'}(x_{1:T}|y_{1:T})\, dx_{1:T} \\ &\qquad + \{ \nabla_\theta \log p_\theta(y_{1:T})|_{\theta=\theta'} \} \{ \nabla_\theta \log p_\theta(y_{1:T})|_{\theta=\theta'} \}^\top \,, \end{aligned} \quad \text{(C.14)}$$

which completes the proof. ■

Bibliography

[1] J.C. Agüero, W. Tang, J.I. Yuz, R. Delgado, and G.C. Goodwin. Dual time-frequency domain system identification. *Automatica*, 48(12):3031–3041, 2012.

[2] B.D.O. Anderson and J.B. Moore. *Optimal filtering*. Prentice Hall, Englewood Cliffs, N.J., 1979.

[3] X. Bombois, G. Scorletti, M. Gevers, P.M.J. Van den Hof, and R. Hildebrand. Least costly identification experiment for control. *Automatica*, 42(10):1651–1662, October 2006.

[4] S. Boyd, P. Diaconis, and L. Xiao. Fastest mixing Markov chain on a graph. *SIAM Review*, 46(4):667–689, October 2004.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[6] C. Brighenti. On input design for system identification: input design using Markov chains, M.Sc. Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2009.

[7] C. Brighenti, B. Wahlberg, and C.R. Rojas. Input design using Markov chains for system identification. In *Joint 48th Conference on Decision and Control and 28th Chinese Conference*, pages 1557–1562, Shangai, P.R. China, 2009.

[8] O. Cappé, E. Moulines, and T. Rydén. *Inference in Hidden Markov Models*. Springer, 2005.

[9] G. Casella and R.L. Berger. *Statistical inference*. Duxbury Pacific Grove, CA, 2002.

[10] M. Casini, A. Garulli, and A. Vicino. Input design for worst-case system identification with uniformly quantized measurements. In *15th IFAC Symposium on System Identification (SYSID)*, Saint-Malo, France, 2009.

[11] M. Casini, A. Garulli, and A. Vicino. Input design in worst-case system identification using binary sensors. *IEEE Transactions on Automatic Control*, 56(5):1186–1191, 2011.

[12] M. Casini, A. Garulli, and A. Vicino. Input design in worst-case system identification with quantized measurements. *Automatica*, 48(5):2297–3007, 2012.

[13] B. Chen and P.K. Willett. Channel optimized binary quantizers for distributed sensor networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 3, pages (iii) 845–848. IEEE, 2004.

[14] D.R. Cox. *Planning of experiments*. New York: Wiley, 1958.

[15] H. Cramér. *Mathematical methods of statistics*. Princeton University Press, Princeton, 1946.

[16] J. Dahlin, F. Lindsten, and T.B. Schön. Second-order particle MCMC for Bayesian parameter inference. In *Proceedings of the 19th IFAC World Congress*, Cape Town, South Africa, August 2014.

[17] N.G. de Bruijn and P. Erdos. A combinatorial problem. *Koninklijke Nederlandse Akademie v. Wetenschappen*, 49:758–764, 1946.

[18] A. De Cock, M. Gevers, and J. Schoukens. A preliminary study on optimal input design for nonlinear systems. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, Florence, Italy, 2013.

[19] P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B*, 68(3):411–436, 2006.

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

[21] J.L. Doob. *Stochastic processes*. New York Wiley, 1953.

[22] A. Doucet and A. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In D. Crisan and B. Rozovsky, editors, *The Oxford Handbook of Nonlinear Filtering*. Oxford University Press, 2011.

[23] V.V. Fedorov. *Theory of optimal experiments*. New York: Academic Press, 1972.

[24] R.A. Fisher. On an absolute criterion for fitting frequency curves. *Messenger of Mathematics*, 41(8):155–160, 1912.

[25] R.A. Fisher. Theory of statistical estimation. *Mathematical Proceedings of the Cambridge Philosophical Society*, 22(5):700–725, 1925.

[26] M. Forgione, X. Bombois, P.M.J. Van den Hof, and H. Hjalmarsson. Experiment design for parameter estimation in nonlinear systems based on multilevel excitation. In *Proceedings of the European Control Conference (ECC)*, Strasbourg, France, 2014.

[27] U. Forssell and L. Ljung. Closed-loop identification revisited. *Automatica*, 35(7):1215–1241, 1999.

[28] K.F. Gauss. Theoria motus corporum celestium, English translation: Theory of the Motion of the Heavenly Bodies. *Dover, New York*, 1963.

[29] L. Gerencsér, H. Hjalmarsson, and J. Mårtensson. Identification of ARX systems with non−stationary inputs: asymptotic analysis with application to adaptive input design. *Automatica*, 45(3):623–633, 2009.

[30] M. Gevers and L. Ljung. Optimal experiment designs with respect to the intended model application. *Automatica*, 22(5):543–554, 1986.

[31] B.I. Godoy, G.C. Goodwin, J.C. Agüero, D. Marelli, and T. Wigren. On identification of FIR systems having quantized output data. *Automatica*, 46(9):1905–1915, 2011.

[32] G.C. Goodwin, J.C. Murdoch, and R.L. Payne. Optimal test signal design for linear SISO system identification. *International Journal of Control*, 17(1):45–55, 1973.

[33] G.C. Goodwin and R.L. Payne. *Dynamic System Identification: Experiment Design and Data Analysis*. Academic Press, New York, 1977.

[34] R.B. Gopaluni, T.B. Schön, and A.G. Wills. Input design for nonlinear stochastic dynamic systems - A particle filter approach. In *18th IFAC World Congress*, Milano, Italy, 2011.

[35] M.C. Grant and S.P. Boyd. *The CVX users' guide*. CVX Research, Inc., 2nd. edition, January 2013.

[36] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[37] R. Hildebrand and M. Gevers. Identification for control: optimal input design with respect to a worst-case ν-gap cost function. *SIAM Journal on Control and Optimization*, 41(5):1586–1608, March 2003.

[38] R. Hildebrand and G. Solari. Closed-loop optimal input design: The partial correlation approach. In *15th IFAC Symposium on System Identification*, Saint-Malo, France, July 2009.

[39] H. Hjalmarsson. System identification of complex and structured systems. *European Journal of Control*, 15(3):275–310, 2009.

[40] H. Hjalmarsson. System identification of complex and structured systems. In *Proceedings of the European Control Conference*, Budapest, Hungary, 2009.

[41] H. Hjalmarsson and H. Jansson. Closed loop experiment design for linear time invariant dynamical systems via LMIs. *Automatica*, 44(3):623–636, 2008.

[42] H. Hjalmarsson and J. Mårtensson. Optimal input design for identification of non-linear systems: Learning from the linear case. In *American Control Conference*, pages 1572–1576, New York, United States, 2007.

[43] R. Horn and C. Johnson. *Matrix Analysis*. Cambridge University Press (9th reprint), 1999.

[44] H. Jansson. *Experiment design with applications in identification for control*. Ph.D. thesis, KTH Royal Institute of Technology, December 2004.

[45] H. Jansson and H. Hjalmarsson. Input design via LMIs admitting frequency-wise model specifications in confidence regions. *IEEE Transactions on Automatic Control*, 50(10):1534–1549, October 2005.

[46] D.B. Johnson. Finding all the elementary circuits of a directed graph. *SIAM Journal on Computing*, 4(1):77–84, March 1975.

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

[48] S. Kar, H. Chen, and P.K. Varshney. Optimal identical binary quantizer design for distributed estimation. *IEEE Transactions on Signal Processing*, 60(7):3896–3901, 2012.

[49] J. Kiefer. General equivalence theory for optimum designs (approximate theory). *The Annals of Statistics*, 2(5):849–879, 1974.

[50] G. Kitagawa and S. Sato. Monte Carlo smoothing and self-organising state-space model. In A. Doucet, N. de Fretias, and N. Gordon, editors, *Sequential Monte Carlo methods in practice*, pages 177–195. Springer, 2001.

[51] A. Kong, J.S. Liu, and W.H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

[52] C. Larsson, M. Annergren, and H. Hjalmarsson. On optimal input design in system identification for model predictive control. In *IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 805–810, Orlando, USA, 2011.

[53] C. Larsson, M. Annergren, H. Hjalmarsson, C.R. Rojas, X. Bombois, A. Mesbah, and P.E. Modén. Model predictive control with integrated experiment design for output error systems. In *Proceedings of the European Control Conference (ECC)*, Zurich, Switzerland, 2013.

[54] C. Larsson, H. Hjalmarsson, and C.R. Rojas. On optimal input design for nonlinear FIR-type systems. In *49th IEEE Conference on Decision and Control*, pages 7220–7225, Atlanta, USA, 2010.

[55] C. Larsson, C.R. Rojas, and H. Hjalmarsson. MPC oriented experiment design. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011.

[56] Y. Lin, B. Chen, and B. Suter. Robust binary quantizers for distributed detection. *IEEE Transactions on Wireless Communications*, 6(6):2172–2181, 2007.

[57] K. Lindqvist and H. Hjalmarsson. Optimal input design using linear matrix inequalities. In *IFAC Symposium on System Identification*, Santa Barbara, California, USA, July 2000.

[58] L. Ljung. Convergence analysis of parametric identification methods. *IEEE Transactions on Automatic Control*, 23(5):770–783, 1978.

[59] L. Ljung. *System Identification. Theory for the User, 2nd ed.* Upper Saddle River, NJ: Prentice-Hall, 1999.

[60] L. Ljung. Prediction error estimation methods. Technical report, Linköping University, Department of Electrical Engineering, October 2001.

[61] L. Ljung and B. Wahlberg. Asymptotic properties of the least-squares method for estimating transfer functions and disturbance spectra. *Advances in Applied Probability*, 24(2):412–440, 1992.

[62] T.A. Louis. Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 44(2):226–233, 1982.

[63] X. Lv and P. Luo. A design of autonomous tracing in intelligent vehicle based on photoelectric sensor. *Yadian yu Shengguang*, 33(6):939–942, 2011.

[64] J.M. Maciejowski. *Predictive Control with Constraints.* Edinburgh Gate, Harlow, Essex, England: Prentice Hall, 2002.

[65] D. Marelli, K. You, and M. Fu. Identification of ARMA models using intermittent and quantized output observations. *Automatica*, 49(2):360–369, 2013.

[66] G. McLachlan and T. Krishnan. *The EM algorithm and extensions.* John Wiley & Sons, 2008.

[67] E.A. Nadaraya. On estimating regression. *Theory of Probability & its Applications*, 9(1):141–142, 1964.

[68] J. Olsson, O. Cappé, R. Douc, and E. Moulines. Sequential Monte Carlo smoothing with application to parameter estimation in nonlinear state space models. *Bernoulli*, 14(1):155–179, 2008.

[69] A. Papoulis. *Probability, random variables, and stochastic processes*. McGraw-Hill, 1984.

[70] M.K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

[71] E. Ramsden. *Hall-effect sensors: theory and application*. Newnes, 2011.

[72] R.T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[73] C.R. Rojas, H. Hjalmarsson, L. Gerencsér, and J. Mårtensson. An adaptive method for consistent estimation of real-valued non-minimum phase zeros in stable LTI systems. *Automatica*, 47(7):1388–1398, 2011.

[74] C.R. Rojas, J.S. Welsh, G.C. Goodwin, and A. Feuer. Robust optimal experiment design for system identification. *Automatica*, 43(6):993–1008, June 2007.

[75] B. Sanchez, C.R. Rojas, G. Vandersteen, R. Bragos, and J. Schoukens. On the calculation of the Data-optimal multisine excitation power spectrum for broadband impedance spectroscopy measurements. *Measurement Science and Technology*, 23(8):1–15, 2012.

[76] T.B. Schön and F. Lindsten. *Learning of dynamical systems – Particle filters and Markov chain methods*. 2014.

[77] T.B. Schön, A. Wills, and B. Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

[78] T. Söderström and P. Stoica. *Instrumental variable methods for system identification*, volume 161. Springer-Verlag, Berlin, 1983.

[79] T Söderström and P Stoica. *System Identification*. Prentice Hall, New York, 1989.

[80] A.J. Sörensen, S.I. Sagatun, and T.I. Fossen. Design of a dynamic positioning system using model-based control. *Control Engineering Practice*, 4(3):359–368, 1996.

[81] J.C. Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, 45(10):1839–1853, 2000.

[82] J.C. Spall. Improved methods for Monte Carlo estimation of the Fisher Information Matrix. In *IEEE American Control Conference*, pages 2395–2400, 2008.

[83] H. Suzuki and T. Sugie. On input design for system identification in time domain. In *Proceedings of the European Control Conference*, Kos, Greece, July 2007.

[84] R. Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*, 1(2):146–160, June 1972.

[85] S.J. Taylor. *Asset price dynamics, volatility, and prediction*. Princeton university press, 2011.

[86] P.E. Valenzuela, J. Dahlin, C.R. Rojas, and T.B. Schön. A graph/particle-based method for experiment design in nonlinear systems. In *the 19^{th} IFAC World Congress*, Cape Town, South Africa, March 2014.

[87] P.E. Valenzuela, C.R. Rojas, and H. Hjalmarsson. Optimal input design for non-linear dynamic systems: a graph theory approach. In *52^{th} IEEE Conference on Decision and Control (CDC)*, Florence, Italy, December 2013.

[88] J.J. Van de Beek, O. Edfors, M. Sandell, S.K. Wilson, and P. Ola Borjesson. On channel estimation in OFDM systems. In *45^{th} IEEE Vehicular Technology Conference*, volume 2, pages 815–819, 1995.

[89] P.M.J. Van den Hof. Closed-loop issues in system identification. *Annual Reviews in Control*, 22:173–186, 1998.

[90] P.M.J. Van den Hof and R.A. de Callafon. Multivariable closed-loop identification: from indirect identification to dual-Youla parametrization. In *Proceedings of the 35^{th} IEEE Conference on Decision and Control*, Kobe, Japan, 1996.

[91] P.M.J. Van den Hof and R.J.P. Schrama. An indirect method for transfer function estimation from closed loop data. *Automatica*, 29(6):1523 – 1527, 1993.

[92] P.M.J. Van den Hof and R.J.P. Schrama. Identification and control: closed-loop issues. *Automatica*, 31(12):1751–1770, 1995.

[93] P. Van Overschee and B. De Moor. Subspace identification for linear systems: theory, implementation, applications. *Kluwer academic publishers*, 1996.

[94] T.L. Vincent, C. Novara, K. Hsu, and K. Poola. Input design for structured nonlinear system identification. In *15th IFAC Symposium on System Identification*, pages 174–179, Saint-Malo, France, 2009.

[95] T.L. Vincent, C. Novara, K. Hsu, and K. Poolla. Input design for structured nonlinear system identification. *Automatica*, 46(6):990–998, 2010.

[96] B. Wahlberg, H. Hjalmarsson, and P. Stoica. On optimal input signal design for frequency response estimation. In *49th Conference on Decision and Control*, pages 302–307, Atlanta, USA, 2010.

[97] A. Wald. Note on the consistency of the maximum likelihood estimate. *The Annals of Mathematical Statistics*, 20(4):595–601, 1949.

[98] L.Y. Wang, G.G. Yin, J-F. Zhang, and Y. Zhao. *System Identification with Quantized Observations*. Birkhäuser, 2010.

[99] T. Wu and Q. Cheng. One-bit quantizer design for distributed estimation under the minimax criterion. In *IEEE 71st Vehicular Technology Conference (VTC)*, pages 1–5. IEEE, 2010.

[100] A. Zaman. Stationarity on finite strings and shift register sequences. *The Annals of Probability*, 11(3):678–684, August 1983.

[101] Q. Zhang and A. Benveniste. Wavelet networks. *IEEE Transactions on Neural Networks*, 3(6):889–898, 1992.