UNIVERSITY OF TWENTE.

Control design for a chemical disinfection process using the residence time distribution

Diana Ugryumova

Master Thesis
Department of Applied Mathematics
Mathematical Theory of Systems and Control Group
The University of Twente, The Netherlands

Supervisor: Hans Zwart

Date of report
March 23, 2010

UNIVERSITY OF TWENTE.

General information

Written by Diana Ugryumova

Master Thesis
At the Mathematical Theory of Systems and Control Group
Department of Applied Mathematics
The University of Twente, The Netherlands

Title project:

Control design for a chemical disinfection process
using the residence time distribution

Supervisor:

Hans Zwart (University of Twente, the Netherlands)

Address:

The University of Twente
Postbus 217
7500 AE Enschede
The Netherlands

Foreword

This is the final report of a Master assignment in Applied Mathematics, at the chair of Mathematical Theory of Systems and Control at the University of Twente, the Netherlands. The goal of this Master assignment is to work further on one of the topics of the PhD thesis of Simon van Mourik [14], being the control design for a chemical disinfection process via residence time distribution.

Many tanks to my fellow PhD students for all the fun and interesting discussions during lunches and coffee brakes, especially to my roommate Svetlana Polenkova. Special thanks to Ruud van Damme and Jaroslav Krystul for numerous ideas on how to solve numerical problems; to Gjerrit Meinsma who was patiently answering my questions at any moment and for introducing me to the beautiful world of Asymptote; to Timon Zijnge for all his support and for the readable figures in this report; to my supervisor Hans Zwart who helped me and motivated me during the whole project. And above all my parents and friends for the emotional support.

Summary

The goal of this project is to design a controller for a ultraviolet disinfection system. A disinfection system consists of a tube with a ultraviolet lamp in its center. Some liquid containing bacteria is flowing through the tube. The aim of the controller is to minimize the total energy consumption of the ultraviolet lamp, while keeping the amount of bacteria on the outlet of the system under a given bound.

To achieve this goal we derive a model for our system. This model is based on the residence time distribution of the bacteria inside the tube. The result is a bilinear state-space model with a possible time-delay.

In this report we compare several methods for identification of the residence time distribution. An algorithm based on system realization theory gives a good estimate. Then, a PID and a LQG controller is designed for the linearized system. The performance of both controllers is tested by implementing them on the bilinear model of the system.

For further research we recommend to explore the possibilities of residence time distribution identification using a positive realization theory. Since bacteria concentration are always positive, we are dealing with a positive system. It would also be interesting to design a controller for the bilinear system directly and compare this to the controller designed for the linearized system.

List of abbreviations

ARX	autoregressive exogenous (model)
BT	balanced truncation (method)
ERA	eigensystem realization algorithm
IO	input-output (model)
IR	impulse response
ISO	input-state output (model)
LTI	linear time invariant
LQG	linear quadratic Gaussian (control)
PID	proportional-integral-derivative (control)
RTD	residence time distribution
SV	singular value
UV	ultraviolet

List of symbols

α	parameter of bacteria destruction rate
β	total residence time of all particles, equals $\int_0^\infty \rho(t)dt$
$\rho(t)$	residence time distribution
$C(t)$	bacteria concentration
$\dot{C}(t)$	extinction rate of the bacteria concentration
$C_{\text{in}}(t)$	input concentration of bacteria
$C_{\text{out}}(t)$	output concentration of bacteria
$e(t)$	estimation error
$h(t)$	impulse response
$K(t)$	lamp intensity
k	discrete time
t	continuous time
$u(t)$	the input
$y(t)$	the output
$\hat{y}(t)$	estimate of the output $y(t)$

Contents

Chapter 1

Introduction

In chemical processes it is often necessary to control the concentration of some particles. The aim of this project is to control the final concentration of bacteria that leave a disinfection reactor. Inside the reactor the bacteria are subjected to the Ultraviolet (UV) light radiation. It is experimentally proved that bacteria are being destroyed when subjected to UV light [8]. At this moment it often happens that the UV light inside the reactor is at its maximum intensity. In other words, the lamp can be either turned on or turned off. With this research we want to explore the possibility of using less than the maximum energy of the UV lamp to destroy a sufficient amount of bacteria.

Chemical disinfection processes play an important role in our everyday life. A simple example of such a process is the disinfection of drinking water. This is usually done using chlorine to ensure the microbiological safety [1]. Because it is a highly reactive gas, the disinfection using chlorine has many by-products, which may be harmful for the environment. There are various alternatives and environment friendlier ways of water disinfection, such as heat, oxidation, filtration or ultraviolet light emission. All these disinfection methods either kill or deactivate the bacteria, therefore preventing it from reproduction.

In our project we use a UV-lamp to reduce the bacteria concentration in apple juice, for previous research see [14]. The disinfection process using UV-light has several advantages: it does not effect the taste, color and smell of the substance that is being disinfected. That is why it is very attractive to use UV-light in disinfecting food and drinks. A clear disadvantage of

[1] In most countries around the world, however not in the Netherlands.

this method is that the performance of the disinfection is reduced by a high turbidity of the liquid and strongly affected by the type, age and density of bacteria. We would like to point out that the UV-light disinfection method can also be used for other disinfection purposes, like the air in a surgery room or for drinking water.

To control a system first we need to model and to identify it. One way to find the model is by looking at the physical characteristics of the system, such as the motion of the fluid inside the reactor, described by the Navier-Stokes equations. In this report we look at the problem from a mathematical point of view, using the experimental input-output measurements of the system as a basis for our model. With this mathematical model we want to describe the essential dynamics of the system. This gives us a relatively simple, though nonlinear, model to work with.

Building a mathematical model of our system, we use the residence time distribution approach. The residence time is approximately the time that a particle stays in the disinfection reactor. Chemical processes have the desirable property that we can measure the residence time distribution of the system. Therefore we can use system identification methods to estimate the system associated with the residence time distribution. The most challenging aspect for the system identification in this project is the positivity of the system, since the bacteria concentrations and the lamp intensity are always positive quantities.

The *final goal* of this project is to design a controller which makes the outgoing concentration small, or smaller than some predefined value, with high certainty. At the same time it is desirable to minimize the energy consumption of the lamp. For controller design purpose the positivity and the nonlinearity of the system are the biggest challenges.

1.1 Report structure

In the beginning of this chapter we have explained the motives for designing a controller for a disinfection system. First, we need to find a suitable model for our disinfection system. The mathematical model based on the residence time distribution of the particles inside the reactor is introduced in Chapter 2. There the assumptions are stated for the mathematical input to output model of the reactor. Then the input-output model is rewritten in a input-state-output form, which is easier to implement. Because this model is nonlinear, a linearization around the equilibria points of the system is performed. Finally,

a model with a time-delay factor is also derived and linearized.

In Chapter 3, we identify the residence time distribution of the particles in a tank. We perform the identification using four different methods. Some of these methods do not work as good as we had expected. Some reasons for these problems are provided. The different methods are then compared with each other.

Chapter 4 treats the controller design for the linearized model derived in Chapter 2. We use two different methods of designing a controller. In Section 4.1 a robust controller is designed using classical control design rules. Using modern control theory we design a LQG controller in Section 4.2. In Section 4.3 we compare the performance of both controllers on the nominal bilinear model of the system.

The conclusions and final remarks are presented in Chapter 5.

At the end of this report an Appendix with some background information about the topics discussed in the report is added. Mathematical background is presented in Appendix A. Extra information about the maximization of the likelihood function from Chapter 3 is found in Appendix B.

To perform the identification of the system and to design a controller the mathematical programming language Matlab [10] is used. A typesetting program LaTeX is used to write this report.

Chapter 2

Mathematical model

In this chapter we derive the mathematical model of the disinfection system. In Section 2.1 we derive an Input-Output (IO) model of the system and state the assumptions. First this is done when the UV lamp is off and then for the case when the UV lamp is on. In Section 2.2 we write the nonlinear IO equation in a bilinear Input-State-Output (ISO) form which is easier for control design implementation. This bilinear ISO model is linearized in Subsection 2.2.3. In Section 2.3 a model with time-delay is derived and linearized in Subsection 2.3.1.

2.1 The Input-Output model derivation

A schematic representation of the disinfection reactor is shown in Fig. 2.1. Here, C_{in} is the concentration of bacteria in the apple juice flowing into the

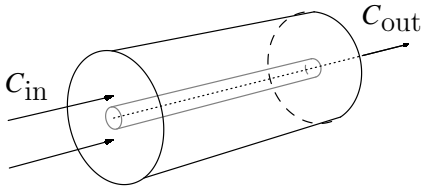


Figure 2.1: Schematic representation of a UV reactor.

cylindric reactor. Inside the reactor the grey tube represents the UV-lamp. When bacteria enter the reactor they are exposed to the UV light irradiation which destroys them. The bacteria that were not destroyed by the UV light

leave the reactor after some time, usually a few seconds. C_{out} represents the concentration of bacteria flowing out of the system.

In this section we derive the mathematical model (see also [14]) of the disinfection system. We consider two different stages of modeling for this particular system: first, when the UV-lamp is off and secondly, with UV-lamp on.

2.1.1 UV lamp off

When the lamp is off, there is no external influence on the bacteria inside the tank. We define the input and the output of the disinfection system as the in- and outflow of the bacteria concentrations in and out of the reactor, respectively. We make the following assumptions about the system:

A-1. *No reaction is taking place*: the total input into the tank is equal to the total output.

A-2. *The relationship between the input and the output is linear*: if you put twice as many bacteria into the tank, then twice as many bacteria will come out of the tank.

The bacteria are living organisms that can grow and reproduce very fast. The linearity assumption is good only if we assume that the bacteria stay inside the reactor not long enough to grow or reproduce.

A-3. *The system is time invariant*: the (chemical) conditions of the tank (the whole system) don't change over time. In other words, it doesn't matter for the results if you do the experiment today or tomorrow.

A-4. *The system is causal*: the number of bacteria coming out of the tank is zero as long as there are no bacteria coming into the tank.

A-5. *The system is positive*: if the input into the system is non-negative signal, then the output will also be non-negative (for a formal definition see Appendix A.1).

We cannot have a negative amount of bacteria on the input nor on the output.

Thus, we assume (assumptions A-2.–A-3.) that our system is Linear Time Invariant (LTI). Intuitively, we can imagine that the particles put inside the tank at one instant, say at time $t = 0$, will come out of the tank at various

times $t > 0$. Thus the particles will have different residence times inside the system [6].

In most chemical processes it is possible to measure the Residence Time Distribution (RTD) - the distribution in time of particles on the output of the system. The shape of the RTD curve depends strongly on the mixing that occurs inside the reactor [6]. For example, a reactor that consists of well-mixed tanks connected in series the RTD has a bell-shaped form.

The RTD can be estimated as follows. During a short fixed time interval a high concentration of some substance (say, ink) is injected at the inlet of the system. On the outlet of the reactor the concentration of this substance is measured, until the concentration becomes zero. For a schematic representation of this phenomenon see Fig. 2.2.

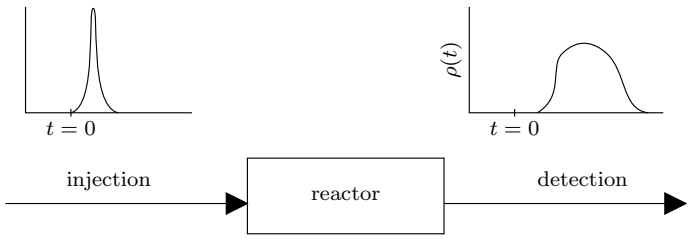


Figure 2.2: A schematic representation of the measurement of the residence time distribution, with an impulse on the input side and on the output the residence time distribution curve $\rho(t)$.

Using assumptions A-2.–A-4., we can write the relation between the input and the output concentration as a convolution (for a proof see Appendix A.4):

$$y(t) = \int_{-\infty}^{t} \rho(t-\tau)u(\tau)d\tau, \tag{2.1}$$

where $u(t)$ and $y(t)$ are the input and output concentrations of bacteria, respectively, and $\rho(t)$ is the RTD. From Eq. (2.1) we see that the RTD function is nothing else than the Impulse Response (IR) of the system [1].

2.1.2 UV lamp on

Once we turn on the UV lamp inside the reactor, a chemical process starts. The energy from the UV light radiation penetrates the bacteria, inactivates

[1] In the literature the IR function is sometimes indicated by $h(t)$.

them by damaging their DNA and thereby prevents them from reproduction. For figures and some statistics see [13]. If exposed to the UV light radiation long enough the bacteria are killed and thus the bacteria concentration is reduced.

Our system has now two inputs, the input concentration into the reactor and the UV-lamp intensity, and one output, the bacteria concentration on the outlet of the reactor. We revise the assumptions made in Subsection 2.1.1 and adjust them (indicated with *):

A*-1. *Reaction is taking place*: the UV-lamp is on which eventually kills the bacteria inside the reactor. We assume that the bacteria extinction rate is uniform through the reactor and is first-order. This would surely be a good assumption when we use a so-called thin film reactor [8], where the liquid substance is pumped as a thin layer through the reactor with (multiple) UV lamp(s) inside the reactor.

A*-2. *The relationship between the input and the output is nonlinear*: because the concentration of bacteria is decreasing inside the reactor tank, we have to consider the rate of extinction of the bacteria to describe the system. The extinction rate is assumed to be proportional to the lamp intensity. When the lamp is off, Subsection 2.1.1, the lamp intensity is zero and the extinction rate of bacteria is zero as well.

Assumptions A-3.–A-5. stay the same as in Subsection 2.1.1.

First we define the symbols used in the model derivation. We are interested in the rate of extinction of the bacteria. In that case we consider the concentrations $C_{\text{in}}(t)$ and $C_{\text{out}}(t)$ as the input and the output of the system, respectively. The UV lamp intensity $K(t)$ is the other input into the system. Later on we are going to control this input lamp intensity $K(t)$ to obtain the desired output concentration $C_{\text{out}}(t)$.

Using assumption A*-1., we propose a first order linear differential equation as a simple model for the extinction rate of the bacteria concentration, $\dot{C}(t)$:

$$\dot{C}(t) = -\alpha K(t) C(t), \tag{2.2}$$

where we assume $K(t) \geq 0$, $\forall t$, α is a certain parameter describing the destruction rate of the bacteria and $C(t)$ is the bacteria concentration per unit volume. The parameter α depends on a lot of factors, such as temperature and chemical characteristics of the fluid, background flora, etc.. Eq. (2.2) applies as long as bacteria are inside the reactor-tube. From the RTD we

know that this time is not fixed. Therefore we combine Eq. (2.1) and (2.2) to get a model for the relationship between the input concentration $C_{\text{in}}(t)$, the output concentration $C_{\text{out}}(t)$ and the lamp intensity $K(t)$.

Assume that bacteria are put into the reactor with the concentration C_{in}, which is constant for some small period of time Δ after time t_0: $C_{\text{in}}(t) = C_{\text{in}}(t_0)$ for $t_0 \leq t \leq t_0 + \Delta$. Assume as well that $C_{\text{in}}(t_0)$ stays inside the reactor for t_r seconds and comes out of the reactor with concentration C_{out,t_r} at time $t_0 + t_r$. We want to calculate this concentration of bacteria, C_{out,t_r}, that are left after being subjected to the UV light inside the reactor. Using Eq. (2.2) we find the following relation between C_{in}, K and C_{out,t_r}:

$$C_{\text{out},t_r}(t_0 + t_r) = C_{\text{in}}(t_0) \exp\left(-\alpha \int_{t_0}^{t_0+t_r} K(\tau)d\tau\right). \tag{2.3}$$

In Eq. (2.3) the integral at the end can be interpreted as the total light that the particles with residence time t_r received during their stay inside the reactor. Substituting $t = t_0 + t_r$ in Eq. (2.3) gives:

$$C_{\text{out},t_r}(t) = C_{\text{in}}(t - t_r) \exp\left(-\alpha \int_{t-t_r}^{t} K(\tau)d\tau\right). \tag{2.4}$$

Equation (2.4) describes the output concentration $C_{\text{out},t_r}(t)$ at any time instant t with $t \geq t_r$ for bacteria with residence time t_r. Given the RTD function $\rho(t)$ of bacteria inside the reactor, we can compute the total output concentration.

Normalizing $\rho(t)$ by the total amount of bacteria that has been inside the reactor $\int_0^\infty \rho(t)dt$, we get a probability density for bacteria with residence time t_r:

$$\frac{\rho(t_r)}{\int_0^\infty \rho(t)dt}.$$

Taking an integral of the product of the output concentration function for bacteria with residence time t_r, $C_{\text{out},t_r}(t)$, with the likelihood that bacteria stay t_r seconds inside the reactor, $\frac{\rho(t_r)}{\int_0^\infty \rho(t)dt}dt_r$, the total output concentration of the bacteria becomes:

$$\begin{aligned} C_{\text{out}}(t) &= \int_0^\infty C_{\text{out},t_r}(t) \frac{\rho(t_r)}{\int_0^\infty \rho(t)dt} dt_r \\ &= \frac{1}{\beta} \int_0^\infty C_{\text{in}}(t - t_r) \exp\left(-\alpha \int_{t-t_r}^{t} K(\tau)d\tau\right) \rho(t_r)dt_r, \end{aligned} \tag{2.5}$$

where $\beta = \int_0^\infty \rho(t)dt$ is a normalizing constant.

Equation (2.5) is the nonlinear mathematical model of our system with UV lamp on. We are going to use this model for the controller design in Chapter 4.

2.2 From IO to ISO (nonlinear) model

In Eq. (2.5), we derived the relation between the output bacteria concentration $C_{\text{out}}(t)$, the input concentration $C_{\text{in}}(t)$, the lamp intensity $K(t)$ and the RTD function $\rho(t)$. For the purpose of designing a controller, this IO equation needs to be rewritten as a state-space model. The ISO model has the advantage that it is easier to implement. To be able to write this IO model in ISO form we assume some representation of ρ as a function of t.

For the model derivations in this and the next sections we assume that all integrals exist.

2.2.1 Scalar form of RTD function

For the purpose of simplicity we first assume that $\rho(t) = \exp(at)$ and we change the coordinates in Eq. (2.5) as $t - t_r = \zeta$. Then:

$$\begin{aligned} C_{\text{out}}(t) &= \frac{1}{\beta}\int_{-\infty}^{t} C_{\text{in}}(\zeta)\exp\left(-\alpha\int_{\zeta}^{t} K(\tau)d\tau\right)\exp\left(a(t-\zeta)\right)d\zeta \\ &= \frac{1}{\beta}\int_{-\infty}^{\infty} C_{\text{in}}(\zeta)\exp\left(-\alpha\int_{\zeta}^{t} K(\tau)d\tau + a(t-\zeta)\right)\mathbb{1}(t-\zeta)d\zeta, \end{aligned} \tag{2.6}$$

where $\mathbb{1}(t-\zeta) = 1$ if $t \geq \zeta$ and 0 elsewhere. It is the well-known indicator function. The last step in Eq. (2.6) is not necessary, but it makes the derivation more clear.

Choosing the state of the system as

$$x(t) = \int_{-\infty}^{t} C_{\text{in}}(\zeta)\exp\left(-\alpha\int_{\zeta}^{t} K(\tau)d\tau\right)\exp\left(a(t-\zeta)\right)d\zeta$$

and taking the derivative of the state gives:

$$\begin{aligned}\dot{x}(t) = \int_{-\infty}^{\infty} \Bigg(& C_{\text{in}}(\zeta) \frac{d}{dt} \left[-\alpha \int_{\zeta}^{t} K(\tau) d\tau + a(t-\zeta) \right] \\ & \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) d\tau + a(t-\zeta) \right) \mathbb{1}(t-\zeta) \\ & + C_{\text{in}}(\zeta) \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) d\tau + a(t-\zeta) \right) \delta(t-\zeta) \Bigg) d\zeta, \\ C_{\text{out}}(t) = & \frac{1}{\beta} x(t),\end{aligned} \tag{2.7}$$

where $\delta(t)$ denotes the Dirac delta function. So we have:

$$\begin{aligned}\dot{x}(t) &= \int_{-\infty}^{t} C_{\text{in}}(\zeta) \left[-\alpha K(t) + a \right] \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) d\tau + a(t-\zeta) \right) d\zeta + C_{\text{in}}(t) \\ &= [-\alpha K(t) + a] x(t) + C_{\text{in}}(t), \\ C_{\text{out}}(t) &= \frac{1}{\beta} x(t).\end{aligned} \tag{2.8}$$

Choosing the input $C_{\text{in}}(t) = u(t)$ and the output $C_{\text{out}}(t) = y(t)$ we find the ISO representation of the IO model in Eq. (2.5):

$$\begin{aligned}\dot{x}(t) &= (-\alpha K(t) + a) x(t) + u(t), \\ y(t) &= \frac{1}{\beta} x(t).\end{aligned} \tag{2.9}$$

We see that the system (2.9) is bilinear in the state, because of the control times state term $K(t) \cdot x(t)$. This result we obtain under the assumption that $\rho(t) = \exp(at)$.

2.2.2 Matrix form of RTD function

In general we can represent the system in a matrix state-space form. Assume that we can express $\rho(t) = Ce^{At}B$. Then the IO model of our system becomes:

$$C_{\text{out}}(t) = \frac{1}{\beta} \int_{0}^{\infty} C_{\text{in}}(t - t_r) \exp\left(-\alpha \int_{t-t_r}^{t} K(\tau) d\tau \right) C \exp(A t_r) B dt_r. \tag{2.10}$$

Note that in our case $C_{\text{in}}(t)$ and $K(t)$ are scalar functions. Changing the coordinates in Eq. (2.10) as $t - t_r = \zeta$, we get:

$$C_{\text{out}}(t) = \frac{1}{\beta} C \int_{-\infty}^{t} \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) I d\tau + A(t-\zeta) \right) B \, C_{\text{in}}(\zeta) d\zeta. \tag{2.11}$$

Choosing the state of the system as

$$x(t) = \int_{-\infty}^{t} \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) I d\tau + A(t-\zeta)\right) B\, C_{\text{in}}(\zeta) d\zeta$$

and taking the derivative of the state with respect to t (this goes in a similar way to equations (2.7) and (2.8)), we get:

$$\begin{aligned}
\dot{x}(t) &= \int_{-\infty}^{t} [-\alpha K(t) I + A] \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) I d\tau + A(t-\zeta)\right) B\, C_{\text{in}}(\zeta) d\zeta \\
&\quad + B\, C_{\text{in}}(t) \\
&= [-\alpha K(t) I + A] x(t) + B\, C_{\text{in}}(t), \qquad (2.12) \\
C_{\text{out}}(t) &= \frac{1}{\beta} C x(t).
\end{aligned}$$

Choosing the input $C_{\text{in}}(t) = u(t)$ and the output $C_{\text{out}}(t) = y(t)$, we get the general (matrix) state space form of the system:

$$\begin{aligned}
\dot{x}(t) &= (-\alpha K(t) I + A) x(t) + B u(t), \\
y(t) &= \frac{1}{\beta} C x(t).
\end{aligned} \qquad (2.13)$$

Thus we have written the IO equation (2.5) in an ISO from in equation (2.13), under the condition that $\rho(t)$ can be written in the from $Ce^{At}B$.

2.2.3 Linearized ISO model

In Eq. (2.13) we found the general ISO representation of our system, under the assumption that $\rho(t) = Ce^{At}B$. This model is bilinear in the state.

When designing a controller one usually starts to design a controller for the linearized system. Then we can compare the performance of the controller for the linearized system to the performance of the controller for the bilinear system. We compare the performance of the controllers by means of the time constant of the system, implementation difficulty, total energy usage, robustness, etc., see Chapter 4.

We perform a standard linearization of the ISO model (2.13) from Section 2.2. Linearizing around the equilibrium point $p^{\text{eq}} = (u^{\text{eq}}, K^{\text{eq}}, y^{\text{eq}}, x^{\text{eq}})$, for which holds

$$0 = (-\alpha K^{\text{eq}} I + A) x^{\text{eq}} + B u^{\text{eq}}, \quad y^{\text{eq}} = \frac{1}{\beta} C x^{\text{eq}}, \qquad (2.14)$$

we get the following, where $x^*(t)$ is the deviation of the state $x(t) = x^*(t)+x^{\text{eq}}$ from the equilibrium x^{eq}:

$$\begin{aligned}\dot{x}(t) &= \frac{d}{dt}(x^*(t) + x^{\text{eq}}) \\ &= (-\alpha(K^*(t) + K^{\text{eq}})I + A)(x^*(t) + x^{\text{eq}}) + B(u^*(t) + u^{\text{eq}}), \\ y(t) &= (y^*(t) + y^{\text{eq}}) = \frac{1}{\beta}C(x^*(t) + x^{\text{eq}}).\end{aligned} \tag{2.15}$$

After rewriting these equations, using the linearization conditions (2.14), and assuming that the nonlinear factor $x^*(t)K^*(t)$ is negligible, we get the linearized ISO form of the system:

$$\begin{aligned}\dot{x}^*(t) &= (-\alpha K^{\text{eq}} I + A)x^*(t) - \alpha x^{\text{eq}} K^*(t) + Bu^*(t), \\ y^*(t) &= \frac{1}{\beta}Cx^*(t).\end{aligned} \tag{2.16}$$

This model (2.16) can be used as an approximation for the original nonlinear model (2.13) around the equilibrium point p^{eq}. The closer we are to the equilibrium point p^{eq}, the better the linearized model (2.16) will approximate (2.13).

2.3 ISO model with time-delay

From Fig. 2.2 we see that there is a time-delay in our system. It can be useful to take the time-delay into account when modeling the system, see Chapter 3. We assume that we can express the RTD function with time-delay $t_d \geq 0$ as $\rho(t) = Ce^{A(t-t_d)}B$ for $t \geq t_d$ and 0 for $t < t_d$. Thus in short notation: $\rho(t) = Ce^{A(t-t_d)}B\mathbb{1}(t - t_d)$. Our IO model with time-delay becomes:

$$C_{\text{out}}(t) = \frac{1}{\beta}\int_0^\infty C_{\text{in}}(t-t_r)\exp\left(-\alpha\int_{t-t_r}^{t} K(\tau)d\tau\right) C\exp(A(t_r-t_d))B\mathbb{1}(t_r-t_d)dt_r.$$

Changing the coordinates as $t - t_r = \zeta$ and putting the matrix C before the integral, we obtain:

$$C_{\text{out}}(t) = \frac{1}{\beta}C\int_{-\infty}^{t}\exp\left(-\alpha\int_{\zeta}^{t} K(\tau)Id\tau + A(t - t_d - \zeta)\right) B\mathbb{1}(t-t_d-\zeta)C_{\text{in}}(\zeta)d\zeta. \tag{2.17}$$

Choosing the state of the system as

$$x(t) = \int_{-\infty}^{t} \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) I d\tau + A(t - t_d - \zeta)\right) B \mathbb{1}(t - t_d - \zeta) C_{\text{in}}(\zeta) d\zeta$$

and taking the derivative of the state $x(t)$ with respect to t, we get:

$$\begin{aligned}
\dot{x}(t) &= \exp(-At_d) B \mathbb{1}(-t_d) C_{\text{in}}(t) + \int_{-\infty}^{t} [-\alpha K(t) I + A] \\
&\quad \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) I d\tau + A(t - t_d - \zeta)\right) B\, \mathbb{1}(t - t_d - \zeta)\, C_{\text{in}}(\zeta) d\zeta \\
&\quad + \int_{-\infty}^{t} \exp\left(-\alpha \int_{\zeta}^{t} K(\tau) I d\tau + A(t - t_d - \zeta)\right) B\, \delta(t - t_d - \zeta) C_{\text{in}}(\zeta) d\zeta \\
&= [-\alpha K(t) I + A] x(t) + \exp\left(-\alpha \int_{t-t_d}^{t} K(\tau) d\tau\right) B\, C_{\text{in}}(t - t_d), \\
C_{\text{out}}(t) &= \frac{1}{\beta} C x(t).
\end{aligned} \tag{2.18}$$

In comparison to the general IO model without a time-delay, (2.12), we get the extra term $\exp\left(-\alpha \int_{t-t_d}^{t} K(\tau) d\tau\right)$ being a weighted delay of $K(t)$.

Choosing the input $C_{\text{in}}(t) = u(t)$ and the output $C_{\text{out}}(t) = y(t)$, we get the general state space form of the system with time-delay t_d:

$$\begin{aligned}
\dot{x}(t) &= (-\alpha K(t) I + A) x(t) + \exp\left(-\alpha \int_{t-t_d}^{t} K(\tau) d\tau\right) B u(t - t_d), \\
y(t) &= \frac{1}{\beta} C x(t).
\end{aligned} \tag{2.19}$$

We see that the output of the model (2.19) is 0 for $t < t_d$, provided the initial state $x(0) = 0$. This is exactly what we wanted to achieve. In comparison to the model without a time-delay, (2.13), we notice that the model of the system with a time-delay, (2.19), has an extra nonlinear term caused by the second term in the differential equation of the state.

In the next chapter we will see that a model with a time-delay gives better estimate for the RTD function, so that we can use this model to design a control later on.

2.3.1 Linearization of the model with a time-delay

Again we perform a standard linearization, this time for the ISO model with time-delay (2.19) found in Section 2.3. For the equilibrium point $p^{\text{eq}} = (u^{\text{eq}}, K^{\text{eq}}, y^{\text{eq}}, x^{\text{eq}})$ holds

$$0 = (-\alpha K^{\text{eq}} I + A)x^{\text{eq}} + \exp\left(-\alpha K^{\text{eq}} t_d\right) B u^{\text{eq}}, \quad y^{\text{eq}} = \frac{1}{\beta} C x^{\text{eq}}. \tag{2.20}$$

Linearizing the model with time-delay around the equilibrium point p^{eq}, with $x(t) = x^*(t) + x^{\text{eq}}$, we get:

$$\begin{aligned}
\dot{x}^*(t) =& (-\alpha K^{\text{eq}} I + A)x^*(t) - \alpha x^{\text{eq}} K^*(t) + (-\alpha K^{\text{eq}} I + A)x^{\text{eq}} - \alpha x^*(t) K^*(t) \\
&+ \exp\left(-\alpha \left[\int_{t-t_d}^{t} K^*(\tau) d\tau + K^{\text{eq}} t_d\right]\right) B(u^*(t - t_d) + u^{\text{eq}}), \\
y(t) =& (y^*(t) + y^{\text{eq}}) = \frac{1}{\beta} C(x^*(t) + x^{\text{eq}}).
\end{aligned}$$

After rewriting these equations, using Eq. (2.20), expanding the exponential $\exp\left(-\alpha \int_{t-t_d}^{t} K^*(\tau) d\tau\right)$, and assuming that the nonlinear factors $x^*(t)K^*(t)$ and $\displaystyle\int_{t-t_d}^{t} K^*(\tau) d\tau$, and the higher order terms of the exponential expansion are negligible, we get the linearized ISO model with a time-delay t_d:

$$\begin{aligned}
\dot{x}^*(t) =& (-\alpha K^{\text{eq}} I + A)x^*(t) - \alpha x^{\text{eq}} K^*(t) \\
&+ \exp\left(-\alpha K^{\text{eq}} t_d\right) \left(-\alpha \int_{t-t_d}^{t} K^*(\tau) d\tau\right) B u^{\text{eq}} \\
&+ \exp\left(-\alpha K^{\text{eq}} t_d\right) B u^*(t - t_d), \\
y^*(t) =& \frac{1}{\beta} C x^*(t).
\end{aligned} \tag{2.21}$$

Furthermore, if we can assume that the time-delay factor t_d is very small, $t_d \ll 1$, or that $K(t)$ varies little in the time-interval $[t - t_d, t]$, then we can approximate the weighted delay of $K(t)$ as $\int_{t-t_d}^{t} K^*(\tau) d\tau \approx t_d K^*(t)$. With these assumptions the Eq. (2.21) can be simplified even more:

$$\begin{aligned}
\dot{x}^*(t) =& (-\alpha K^{\text{eq}} I + A)x^*(t) - \left[\alpha x^{\text{eq}} + \exp\left(-\alpha K^{\text{eq}} t_d\right) \alpha t_d B u^{\text{eq}}\right] K^*(t) \\
&+ \exp\left(-\alpha K^{\text{eq}} t_d\right) B u^*(t - t_d), \\
y^*(t) =& \frac{1}{\beta} C x^*(t).
\end{aligned} \tag{2.22}$$

The models (2.21) and (2.22) can be used as an approximation for the original nonlinear model with time-delay (2.19) around the equilibrium point p^{eq}. The closer we are to the equilibrium point p^{eq}, the better the linearized models (2.21) and (2.22) will approximate (2.19).

As we would expect, with zero time-delay, $t_d = 0$, the linearized models with time-delay (2.21) and (2.22) give us the linearized model without time-delay (2.16).

Chapter 3

Estimation of the residence time distribution

As already mentioned in the Introduction (Chapter 1), the goal of this thesis is to design a controller for a chemical disinfection system. For that we first need to identify the chemical system we are dealing with. The process of building mathematical models of dynamical systems using observed data from the system is called System Identification [11].

In this chapter we are going to explore several methods for the identification of the measured Residence Time Distribution. From practice we know that the RTD function has a bell-shaped form with a heavy tail. Before we are going to perform system identification on an experimental RTD, we want to understand the identification procedure on a known (test) system. For that purpose we are going to generate a test RTD function that has the desired bell-shaped form. We use this test system to compare the performance of several system identification methods. Next, we state and explain some important identification objectives for a disinfection system.

In Section 3.1 a model reduction technique called Balanced Truncation (BT) is presented and treated. In Section 3.2 a method called Eigensystem Realization Algorithm (ERA) is introduced, which is based on Markov parameters of the system. In theory the ERA method is very similar to BT, but in practice the first method gives better results than the second. In Section 3.3 we discuss the parametric system identification method from impulse response data. Unfortunately, this method does not give good results. Explanations why this parametric identification method does not work are given in Subsection 3.3.1. The last system identification method treated in

this report is based on maximization of a certain Likelihood Function, Section 3.4. Of the investigated methods, this is the only one that takes the positivity of our system into account. Thus we expect to get the best results using this last identification method.

Measured RTD

We make a realistic assumption that our reactor is non-ideal - it is not perfectly mixed everywhere. There are different ways to model a non-ideal reactor [6]. For example, by a series of tanks connected together with the same total volume as the initial reactor. For the tanks-in-series model of the reactor, the measured RTD usually has a bell-shaped form as in Figure 3.1. Each of the tanks connected in series is assumed to be ideal, i.e., perfectly mixed.

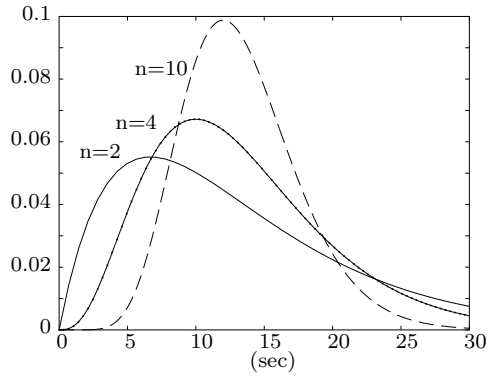


Figure 3.1: Examples of RTD curves. n is the number of well-mixed tanks connected in series.

In modern mathematical systems theory the most common way of describing a dynamical system is a general continuous-time ISO form:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \quad x(0) = x_0, \\ y(t) &= Cx(t) + Du(t), \end{aligned}$$

where t represents continuous time, $u(t)$ is the input, $x(t)$ is the state with x_0 as initial state, $y(t)$ is the output and A, B, C, D are matrices of appropriate dimensions. Note that in Chapter 2 we derived models for our system using this ISO from. The matrix D describes the throughput of the system, i.e., the part of the output that is influenced by the input directly. In our system there are no particles that go directly from the input to the output. That is

why for the disinfection system we can assume that there is no throughput, $D = 0$. We also assume that for negative time the input into the system is zero $u(t) = 0$, $\forall\, t < 0$, therefore the system is initially at rest, $x_0 = 0$. That is why we use the following ISO form to describe our disinfection system:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \quad x(0) = 0, \\ y(t) &= Cx(t). \end{aligned} \tag{3.1}$$

We can write down explicitly the output in terms of the input. However, we are mainly interested in the IR $h(t)$ of the system (3.1):

$$h(t) = C \exp(At) B. \tag{3.2}$$

As we have seen in Subsection 2.1.1, the experimental RTD function is the same as the IR of a system. Hence we can write the experimental RTD function as a continuous-time IR of the form (3.2).

There are several ways of writing a RTD in the (3.2)-form. First of all, because the RTD measurements are in discrete time, we want to write the RTD as an IR in discrete time:

$$h_d(k) = C_d {A_d}^{k-1} B_d, \quad h_d(0) := 0, \quad k = 1 \dots N, \tag{3.3}$$

with matrices A_d, B_d, C_d the discrete-time equivalent of A, B, C in Eq. (3.2) and k the discrete time step. For the continuous-to-discrete time transformation, see Appendix A.3.

Given a discrete-time RTD function $\rho(k-1)$, $k = 1 \dots N$, with $\rho(N-1) \approx 0$, we claim that the following choice of the matrices A_d, B_d and C_d will give us the desired IR (3.3):

$$A_d = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ \vdots & & & \ddots & 1 \\ 0 & \dots & \dots & 0 & 0 \end{pmatrix}, \quad B_d = \begin{pmatrix} \rho(0) \\ \vdots \\ \\ \vdots \\ \rho(N) \end{pmatrix}, \quad C_d = \begin{pmatrix} 1 & 0 & \dots & 0 \end{pmatrix}. \tag{3.4}$$

The proof of the claim is straightforward

$$\begin{aligned} &\text{for } k = 0 \text{ the IR } h_d(0) := 0, \\ &\text{for } k = 1 \text{ the IR } h_d(1) = C_d B_d = \rho(0), \\ &\quad \vdots \\ &\text{for } k = N \text{ the IR } h_d(N) = C_d {A_d}^{N-1} B_d = \rho(N-1). \end{aligned}$$

Thus, putting the RTD measurements in the form (3.4) we get a discrete ISO system of the same order as the number of RTD measurements, i.e., N. We notice that the system of order N is too big to work with, but fortunately there are ways of reducing the order of the system, see Sections 3.1 and 3.2.

Generating a test residence time distribution

Now we generate a test system, which helps us to compare different system identification methods. We assume that the chemical disinfection system can be approximated by a series, say n, of small tanks, see Fig. 3.2, which are all well mixed.

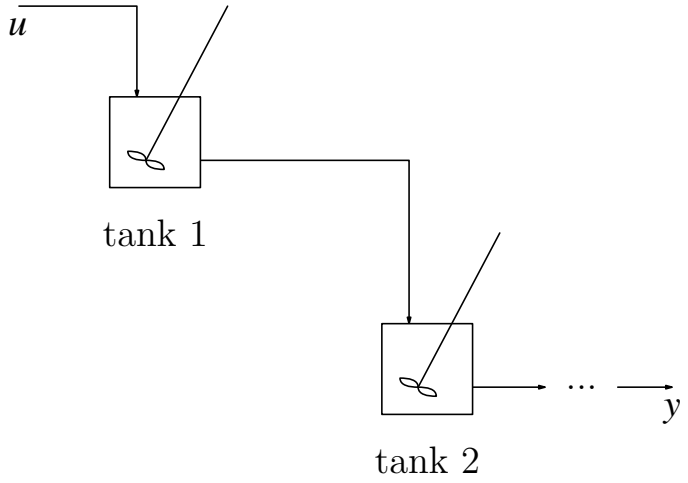


Figure 3.2: A schematic representation of the tanks in series model of the system.

To generate a test RTD function we want to derive a certain ISO form for our system. Assume that the input u enters tank 1, the outflow rate λ of tank 1 is exactly the same as the inflow rate of tank 2, the outflow rate λ of tank 2 is exactly the same as the inflow rate of tank 3, etc.. Define $x_i(t)$ as the concentration of particles in tank i at time t, for $i = 1 \dots n$. We measure the outflow y of the last tank.

We obtain the following ISO form of the system:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t), \end{aligned} \tag{3.5}$$

with the following matrices:

$$A = \begin{pmatrix} -\lambda & 0 & 0 & \dots & 0 \\ \lambda & -\lambda & 0 & \ddots & \vdots \\ 0 & \lambda & -\lambda & \ddots & \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \lambda & -\lambda \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad C = \begin{pmatrix} 0 & \dots & 0 & \lambda \end{pmatrix},$$

where λ is a real positive number, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times 1}$, $C \in \mathbb{R}^{1\times n}$. The Residence Time Distribution $\rho(t)$ is the output of the system if the input u is the pulse.

In the next sections we are going to use the structure (3.5) together with

$$n = 20 \quad \text{and} \quad \lambda = 1.5 \tag{3.6}$$

as our test system. The IR of this test system is evaluated in continuous time and then sampled once a second. It is shown in Figure 3.3. We see that this IR has the desired bell-shaped form.

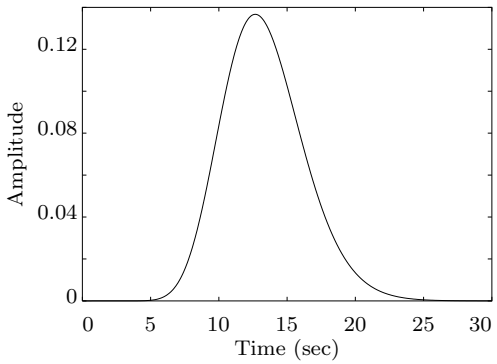


Figure 3.3: Impulse response of our test system.

System identification objectives

There are several system identification objectives to which we want to pay closer attention.

First, we take a closer look at the model equations (2.4) and (2.5) derived in the previous chapter. Putting $t = t_r$ in (2.4), we get the output concentration of particles with residence time t_r at the time instant t_r:

$$C_{\text{out},t_r}(t_r) = C_{\text{in}}(0) \exp\left(-\alpha \int_0^{t_r} K(\tau) d\tau\right). \tag{3.7}$$

From this equation we see that the initial input concentration $C_{\text{in}}(0)$ decays exponentially inside the system, provided the lamp is turned on. The decay holds because we assumed in Eq. (2.2) that the light intensity function $K(t) \geq 0$, $\forall t$, implying that the integral $\int_0^{t_r} K(\tau)d\tau \geq 0$. Notice, that for particles with a small residence time the exponent in (3.7) is approximately 1 and $C_{\text{out},t_r} \approx C_{\text{in}}$. Thus, particles with the smallest residence time contribute to the total output concentration $C_{\text{out}}(t)$ the most, see Eq. (2.5). Therefore we want to model the particles with small residence time properly.

Secondly, we assume that when the UV-lamp is off no reaction is taking place inside the reactor and thus that there is conservation of mass. That means that the total input into the system must be equal to the total output. In fact, because our system is causal and LTI, the conservation of mass directly follows from the total IR being equal to 1, i.e., $\int_0^\infty h(t)dt = 1 \iff \int_0^\infty u(t)dt = \int_0^\infty y(t)dt$. For a proof see Appendix A.5.

Thirdly, the bacteria concentrations in our system are positive. Therefore we are dealing with a positive system, see assumption A-5. in Subsection 2.1.1. From [1, 4] we know that the system is positive if and only if A, B, C are positive matrices. In discrete time this means that all entries of these matrices are non-negative. In continuous time matrix A should be Metzler, i.e., the off-diagonal elements are non-negative. For formal definitions of a positive system and a Metzler matrix, see Appendix A.1.

Summarizing, when identifying our system we want to pay closer attention to the following aspects:

B-1. The estimate of $\rho(t)$ should have a better fit for small t than for large t (by using some kind of weighting function).

B-2. $\int_0^\infty \hat{\rho}(t)dt = 1$, where $\hat{\rho}(t)$ is the estimate of the RTD.

B-3. In discrete time, the system realization matrices A, B and C should all be positive. In continuous time, the matrix A should be Metzler and B, C positive.

Next, we are going to use four different system identification methods. We compare the outcomes of these methods using our test system, (3.5)–(3.6).

3.1 Identification of RTD via balanced truncation

Background

Knowing the IR of the system gives us the opportunity to construct a state-space representation of the system with the dimension equal to the number of measurement points, see the beginning of this chapter. For example, if you have 30 data points, you get a system of order 30. It could be difficult to design a controller and time consuming to simulate the results for a system of such a high order.

That is why we want to explore the possibilities of making the systems order as low as possible, while keeping the essential dynamics of the system as close as possible to the original one. One way of doing this is by performing a Balanced Truncation (BT) [15], which uses the Hankel Singular Values (SV's) of the system, see Appendix A.1 for the definition.

We analyze the system by looking at its Hankel SV's. Intuitively, a Hankel SV is a measure of energy for each state in the system. Physically speaking, a system needs to generate some energy to reach a certain state (controllability of a state). Likewise, there is some amount of energy stored in each state (observability of a state).

The plot of Hankel SV's of our test system is shown in Figure 3.4. We see that there are at most six relevant Hankel SV's and the other values are negligible. The states with small Hankel SV's have relatively low-energy.

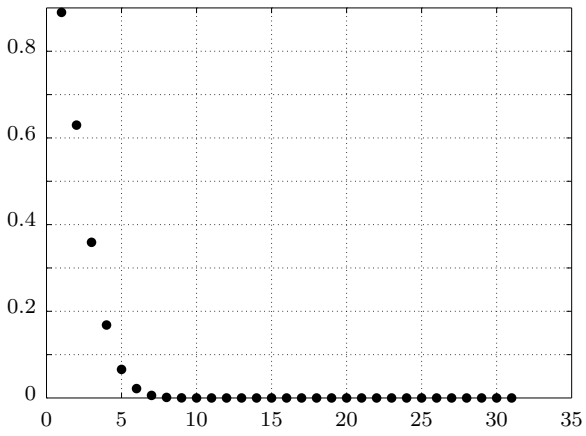


Figure 3.4: The Hankel singular values of the test system.

The idea behind the BT method is that we can discard all relatively small Hankel SV's of the system. These low-energy states can be discarded while keeping the systems dynamics close to the original system [15].

We need a formal definition by which we can discard small Hankel SV's of the system. In system theory one usually wants to minimize some norm of the error.

In this case we consider the error to be the difference in magnitude of the original system and the reduced order system. We want to minimize the following weighted error in the frequency-domain [15]:

$$\min_{order(G_{\text{red}})\leq n} \|W(G - G_{\text{red}})\|_\infty, \tag{3.8}$$

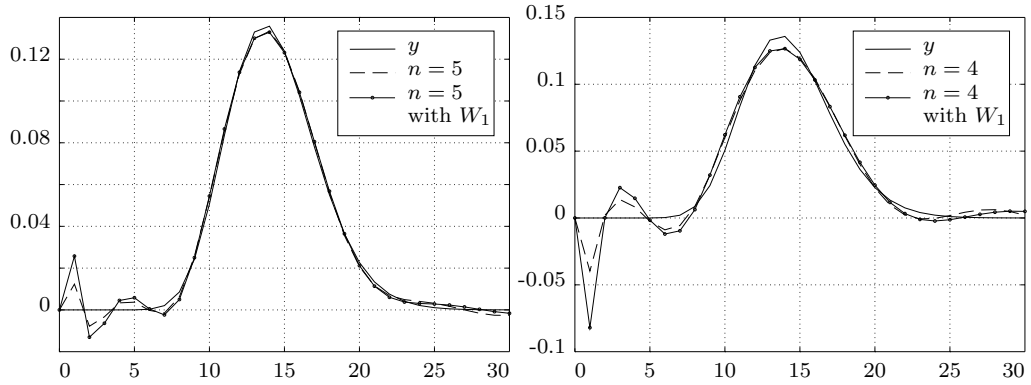
where W is a suitable weighting function in the frequency-domain, G is the transfer function of the original system and G_{red} is the transfer function of the reduced order system; for the definition of H_∞-norm see Appendix A.2. We choose the weighting function W such that there is a higher weight on the particles with small residence time than on particles with large residence time (see the identification objectives in the beginning of this chapter).

Results

We use the function REDUCE in Matlab to perform BT. We compare the results of the model reduction by looking at the IR of G and of G_{red}. We consider G_{red} as a good fit if it's IR shows a good resemblance to the IR of the original system G, especially for small times t.

First, we want to reduce the order of our system by minimizing the norm of Eq. (3.8) without using a weighting function, i.e., $W_0(s) = 1$. From Figure 3.4 we can see that we can discard at least twenty-four SV's. Thus we try to get G_{red} of orders 5 and 4. The resulting IR's are presented in Figure 3.5 (dashed line). Notice, that the scale of the two figures is different. The IR of G_{red} of order 5 gives a better approximation of the original test system, as we would have expected. Both G_{red}'s of order 5 and of order 4 have a large error with respect to the original system G for small time t. Using a suitable weighting function W, we next want to improve the IR of G_{red} for small t.

In the frequency-domain the weighting function objectives are reversed, compared to the time-domain. Therefore we want to put a small weight on small values of s and a large weight on large s. For example a weighting function $W_1(s) = \frac{s+0.001}{0.1s+0.2}$ satisfies our requirements. This gives a small weight



(a) Order G_{red} is 5.

(b) Order G_{red} is 4.

Figure 3.5: The IR of the test system compared to the IR of reduced order system and reduced order system using weighting function $W_1(s)$.

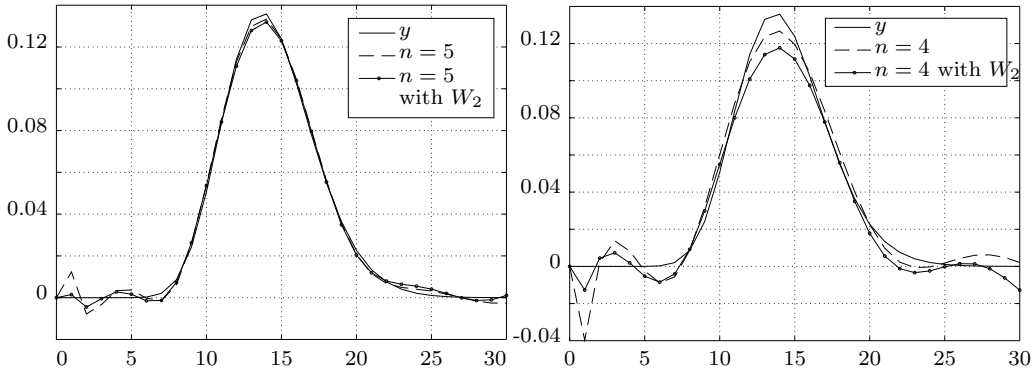
$W_1 = 0.005$ for $s = 0$ and a large weight $W_1 = 10$ for $s \to \infty$, exactly what we wanted.

Again we compare the reduced systems of orders 5 and 4 in Figure 3.5 (circles line). Looking at the IR for small t we see that performing a BT with the weighting function W_1 gives worse results than without weighting, using W_0. This is against our expectations and until now we unfortunately do not have an explanation for this.

For comparison we use the weighting function $W_2(s) = \frac{0.0001s+1}{0.5s+0.1}$. This gives $W_2 = 10$ for $s = 0$ and $W_2 \approx 0$ for large s. In other words, this is exactly opposite the weighting conditions coming from our identification objectives.

We compare the results of the weighting function W_2 to no weighting W_0, Figure 3.6. We see that for small t the IR of the reduced order system using the weighting function W_2 is closer (in magnitude) to the IR of the original system G. But, for large t the deviation from the IR of the nominal system gets bigger, opposite to the results with weighting function W_1.

Let us look at the error norms of the differences between the original and the reduced order systems, in both frequency- and time-domain. The error bounds of the reduced order systems are presented in Table 3.1. We consider the H_∞-norm and the L_1-norm of the errors, see Appendix A.2 for the formal definition of these norms. For the BT method, the H_∞-norm of the error is bounded from above by two times the sum of the left out Hankel



(a) Order G_{red} is 5.

(b) Order G_{red} is 4.

Figure 3.6: Compare the IR of the nominal, reduced and reduced using weighting function $W_2(s)$.

$\downarrow G_{\text{red}}$ order	H_∞ error norm (upper bound)			L_1 error norm ($\approx$)		
weights $\rightarrow$	W_0	W_1	W_2	W_0	W_1	W_2
5	0.06	0.01	0.01	0.07	0.07	0.15
4	0.19	0.05	0.03	0.19	0.19	0.32

Table 3.1: Error bounds of the reduced order systems.

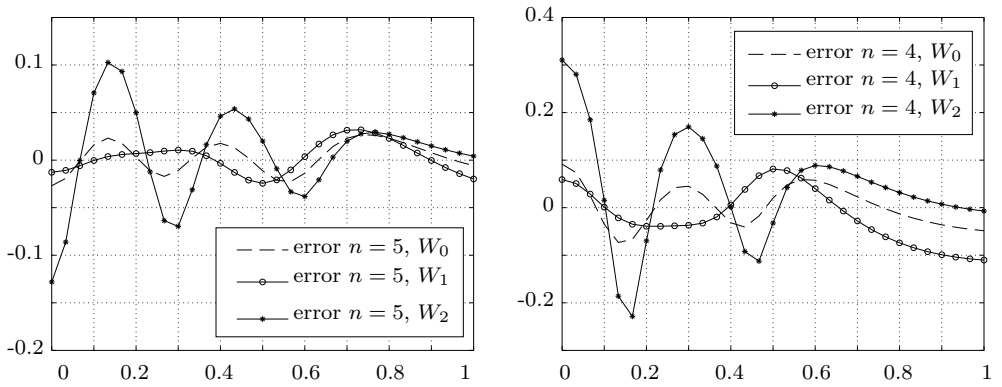
SV's $\sigma_{red+1}, \sigma_{red+2}, \ldots, \sigma_N$:

$$\|G - G_{\text{red}}\|_\infty \leq 2(\sigma_{red+1} + \sigma_{red+2} + \ldots + \sigma_N).$$

The L_1-norm is approximated here by a sum, i.e., it is approximately the difference of the area's under the absolute IR's, see Appendix A.2.

From Table 3.1 we see that the error bound of the reduced order systems in the frequency-domain becomes smaller using weighting functions W_1 and W_2, compared to no weighting W_0. According to the theory, see Appendix A.2, the L_1-norm of the error is larger or equal to the corresponding H_∞-norm of the error.

In Figure 3.7 the magnitude errors of the estimated transfer function G_{red} compared to the transfer function G of the original system are presented. The effects on the estimated G_{red} of using a weighting function W_0, W_1 or W_2 are compared. In Figure 3.7(a) the G_{red}'s of order $n = 5$ and in Figure 3.7(b) the G_{red}'s of order $n = 4$ are presented. We would expect that using the weighting



(a) Errors of the models with order $n = 5$. (b) Errors of the models with order $n = 4$.

Figure 3.7: Magnitude errors in the frequency domain of the estimated models using different weightings W_0, W_1 and W_2.

W_1 would give better estimation results for large s than using the weighting W_2, because the prior has a higher weighting on large s. From both figures of the magnitude estimation errors we see that it is the other way around, i.e., using W_2 seems to give smaller estimation errors for large s than using W_1. As we mentioned before, the results described above are contradictory to our expectations, but until now we do not have an explanation for it.

3.2 Eigensystem Realization Algorithm

Background

In this section we use the Eigensystem Realization Algorithm (ERA) [7] to identify our system with a low-order model. In theory this method is equivalent to the BT system-order reduction method. However, in practice the ERA method usually gives better numerical results than BT method. The ERA method is based on a matrix containing the Markov parameters of the system. Markov parameters are defined as the discrete impulse responses at the time steps k, i.e., the $CA^{(k-1)}B$ terms (throughout this section we assume that the matrices A, B and C belong to a discrete-time system). Construction of the state-space matrices A, B and C is called the system realization problem. In the beginning of this chapter we showed how to construct a state-space model from IR measurements, where the order of the model is equal to the number of measurement points. Thus, the real problem of realization theory is constructing a state-space model with order as low as possible while keeping the dynamics of the reduced order system as close as possible to the original system.

Problem formulation. Given the IR values ρ_k in discrete time, construct a (minimal) state-space realization (A, B, C) in terms of ρ_k, such that $\rho_k = CA^{k-1}B$ holds for every k.

ERA approach. Construct the generalized Hankel $r \times m$ matrix $H(j)$ from the Markov parameters:

$$H(j-1) = \begin{pmatrix} \rho_j & \rho_{j+1} & \cdots & \rho_{j+m-1} \\ \rho_{j+1} & \rho_{j+2} & \cdots & \rho_{j+m} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{j+r-1} & \rho_{j+r} & \cdots & \rho_{j+r+m-2} \end{pmatrix},$$

where r and m are arbitrary integers. In this case we choose $r = m = N$, where N is the number of measurements points of the IR. In this way we get a symmetric matrix with zeros under the anti-diagonal.

Then the Hankel matrices $H(0)$ for $j = 1$ and $H(1)$ for $j = 2$ are used in a clever way to construct the state-space matrices A, B and C of the given system, see [7] for the details. Let us notice that:

$$H(0) = \mathcal{O}_r \mathcal{C}_m \text{ and } H(1) = \mathcal{O}_r A \mathcal{C}_m,$$

where $\mathcal{O}_r = \begin{pmatrix} C & CA & CA^2 & \dots & CA^{r-1} \end{pmatrix}^{\mathrm{T}}$ is the observability matrix and

$\mathcal{C}_m = \begin{pmatrix} B & AB & A^2B & \dots & A^{m-1}B \end{pmatrix}$ is the controllability matrix of the system.

Performing singular value decomposition on the matrix $H(0) = U\Sigma V^{\mathrm{T}}$, the minimal state-space of the given single-input-single-output system is then identified as:

$$A = \Sigma^{-\frac{1}{2}} U^{\mathrm{T}} H(1) V \Sigma^{-\frac{1}{2}} , \quad B = \Sigma^{\frac{1}{2}} V^{\mathrm{T}} E_1 , \quad C = E_1^{\mathrm{T}} U \Sigma^{\frac{1}{2}},$$

with $E_1 = \begin{pmatrix} 1 & 0 & \dots & 0 \end{pmatrix}^{\mathrm{T}}$.

Results

To be able to compare this identification method to the other methods treated in this report, we use the ERA to estimate a reduced order discrete-time model of our test system. In the next section we use autoregressive models to estimate our system. As a result we get a transfer function describing the reduced order system. To be able to compare the ERA result and the results from the next section, we convert the state-space (A, B, C) ERA model to a transfer function.

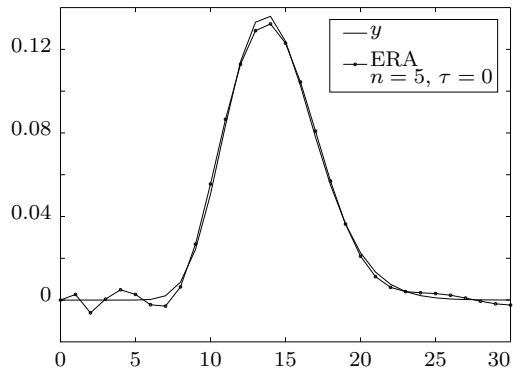
From previous section we know that a system of order 5 gives us a good estimate of our original test system. We also compute ERA estimates for lower orders $n = 4$ and $n = 3$, for comparison.

The IR of the estimated system is compared to the IR of the test system in Figure 3.8. In Figure 3.8(a) the IR of the original test system is compared to the IR of the reduced order system of order 5. We can see that roughly the first 8 seconds of the fit are not as good as we would like - the deviations from the IR of the original system are large.

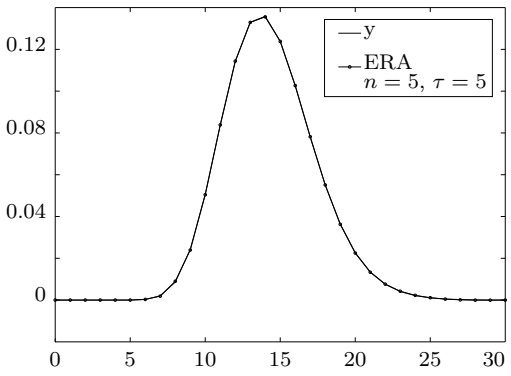
We say that the time starting at zero until the IR of the system deviates significantly from zero is approximately the time-delay of our system. From the IR of our test system in Figure 3.3 we can see that our system has a time-delay τ of approximately 5 seconds. To improve the estimate we can add a time-delay to our model. We do this by setting the first values of the output $y(k)$ to zero and applying again the ERA for $y(k), k > 5$. This indeed gives a better fit for $k > 5$, see Figure 3.8(b).

With the ERA method we get the following transfer function estimate of order $n = 5$ with a time-delay $\tau = 5$.

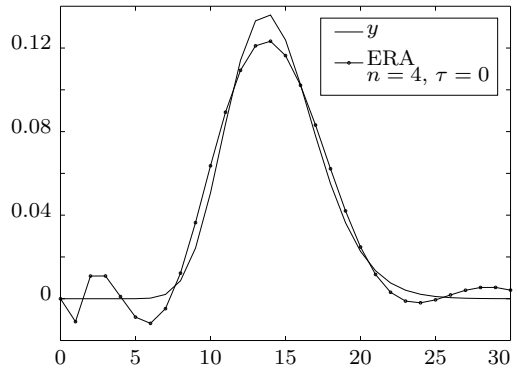
$$\frac{P_N(q)}{P_D(q)} = \frac{q^{-5}(0.0003q^{-1} + 0.0008q^{-2} + 0.0044q^{-3} + 0.0021q^{-4} + 0.0071q^{-5})}{1 - 3.23q^{-1} + 4.40q^{-2} - 3.16q^{-3} + 1.20q^{-4} - 0.20q^{-5}} \tag{3.9}$$



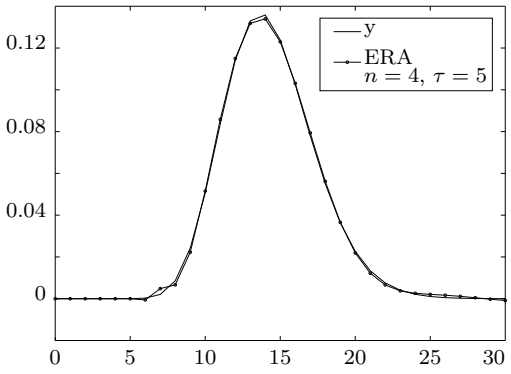
(a) IR estimate using the ERA algorithm, 5-th order.



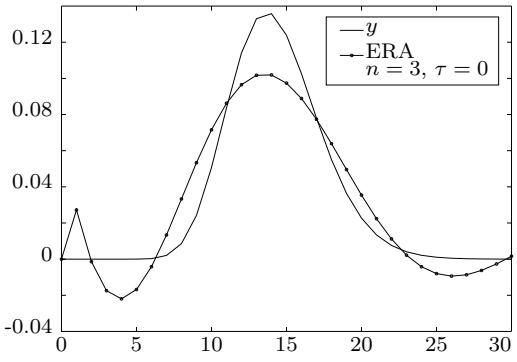
(b) ERA fit with time-delay taken into account - the first 5 entries of the output are set to zero.



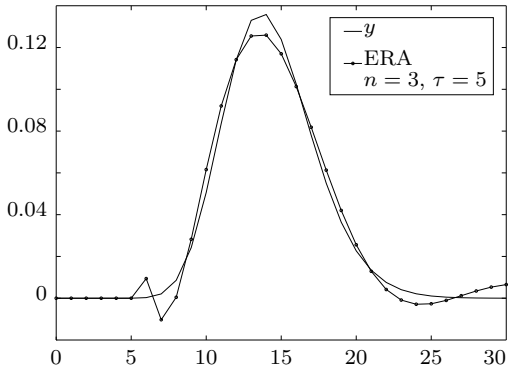
(c) IR estimate using the ERA algorithm, order $n = 4$.



(d) ERA fit with time-delay $\tau = 5$ and order $n = 4$.



(e) IR estimate using the ERA algorithm, order $n = 3$.



(f) ERA fit with time-delay $\tau = 5$ and order $n = 3$.

Figure 3.8: IR's of the fitted transfer functions using ERA.

with poles at

$$\begin{aligned} p_1 &= 0.56 + 0.44i, \\ p_2 &= 0.56 - 0.44i, \\ p_3 &= 0.73, \\ p_4 &= 0.69 + 0.21i, \\ p_5 &= 0.69 - 0.21i. \end{aligned}$$

Because the fifth order gives such good results, we try to reduce the order of the estimated system even more. We do the same procedure for $n = 4$ and $n = 3$. The results are presented in figures 3.8(c) – 3.8(f). As we would expect, the lower the order of the estimate, the worse the result. Although if we account for the time-delay and set the first few values of the IR to zero we still get good results with an estimate of order 4, Figure 3.8(d). For a system estimate of order 3, Figure 3.8(f), we can see relatively large jumps just after 5 seconds. Because we want our estimate to have a better fit for small time t, the minimal order realization of our test system in this case would be $n = 4$, taking a time-delay of the system into account.

Of the two RTD identification methods we already discussed, this method gives the best results. We see that a system of order $n = 4$ already gives a good estimate of our test system. We could even try to use the third order estimate, setting the first 5 or even 6 values to zero.

3.3 Identification of RTD from IR

Background

There is a complete theory about system identification, see [11,12]. Although relatively little research is done about *parametric* system identification from impulse response data. It is straightforward to use the Realization Theory to identify a system from it's IR, but not much is known about fitting some parametric structure to the IR data. The reason for this could be that for most (physical) systems it is difficult or even impossible to measure the IR. For example, in electric networks it is not possible to generate a voltage pulse. Even if we could do that (using the lightning strike), the high voltage pulse input could destroy the network.

Another difficulty we have to cope with is that we have a small number of measurement points to fit a suitable parametric model. For example, the IR

of our test system consists of 30 points. If we would like to fit a parametric structure of order 5 we have to estimate 9 parameters with 30 data points.

In any case we start by fitting some system identification model structures, such as IV4, PEM, AR(MA)X, etc. to our test system (3.5)–(3.6). A big advantage of these models compared to the BT method from Section 3.1 is that we can model the so-called "dead"-time of the system explicitly. "Dead"-time is the delay time of the system. By modeling the dead-time we expect to get better results, because as we can see in Figure 3.3, the IR of our test system is almost zero for $t \leq 5$.

In this section we are going to estimate our test system with the following discrete Autoregressive eXogenous (ARX) model structure:

$$P_D(q)y(k) = P_N(q)u(k-\tau) + e(k), \tag{3.10}$$

where k represents discrete time, q is the time-shift operator, $P_D(q)$ and $P_N(q)$ are polynomials of orders n_{P_D} and n_{P_N}, respectively, with $n_{P_N} \leq n_{P_D}$, τ is the delay of the system and the equation error-term $e(t)$ is assumed to be white noise. We can convert the estimated discrete-time model to continuous-time by using the transformation described in the Appendix A.3.

Results

From Section 3.1 we know that we can approximate our system by a model of order $n \leq 6$. We use the System Identification toolbox of Matlab [10] to fit the discrete ARX structure (3.10).

We can estimate the time-delay of the system from the IR function, the same way we did it in the previous section. Time-delay of the system means that there is a minimum residence time for the bacteria inside the reactor. The time-delay of our system is approximately $\tau = 5$. After a couple of trial-and-error estimations with ARX structures of different orders, we chose $n_{P_N} = 1$, $n_{P_D} = 5$ and the delay-term τ equal to 5 seconds. Then we get the following transfer function:

$$\frac{P_N(q)}{P_D(q)} = \frac{2.1 \cdot 10^{-4} q^{-5}}{1 - 4.18q^{-1} + 7.48q^{-2} - 7.17q^{-3} + 3.68q^{-4} - 0.81q^{-5}} \tag{3.11}$$

with poles at:

$$
\begin{aligned}
p_1 &= 0.70 + 0.65i, \\
p_2 &= 0.70 - 0.65i, \\
p_3 &= 0.96, \\
p_4 &= 0.91 + 0.32i, \\
p_5 &= 0.91 - 0.32i.
\end{aligned}
$$

For the plot of the poles with their 99% confidence intervals, see Figure 3.9. The poles are denoted with a cross. We can see that the confidence regions of the four complex poles are partly outside the unit stability circle. Thus the estimated ARX model could become unstable, due to uncertainties in the model parameters.

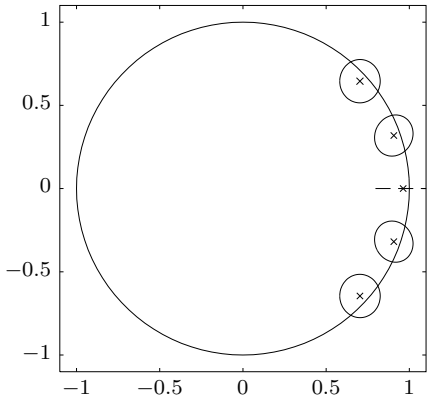


Figure 3.9: The poles with their confidence intervals of the fitted ARX structure.

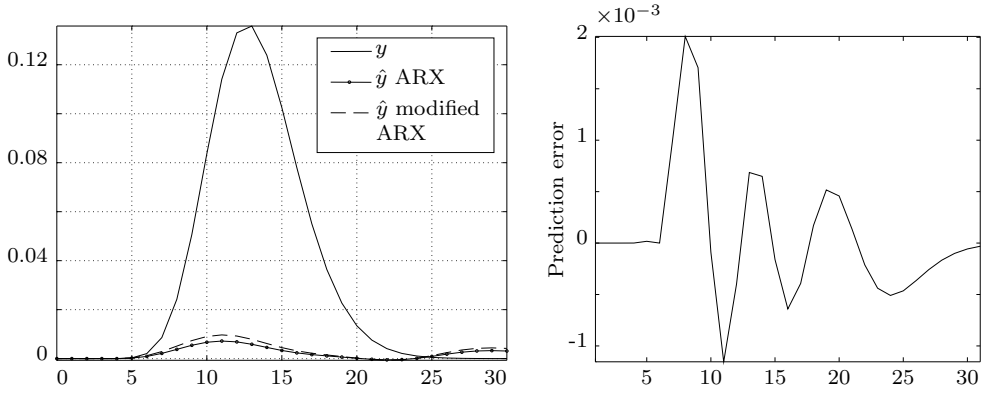
The estimated parameters of this ARX model have huge standard deviations σ, as we can see in Table 3.2. For the numerator of the estimated transfer function, $P_N(q)$, σ_{P_N} is even bigger than the value of the parameter itself!

In Figure 3.10(a) the IR of the original (sampled continuous-time) system is compared to the IR of the estimated ARX model (dashed line). We can see that the estimated model is an awful fit to our test system.

In Figure 3.10(b) the prediction errors $y(k) - \hat{y}(k)$ (with $\hat{y}(k)$ the estimated output) of the estimated ARX model are presented. Even for such a small data-set of 30 points, the prediction errors look like white noise. There is the following relation between the prediction error and the equation error of the

parameter	value	σ
$P_{D,1}$	-4.18	0.13
$P_{D,2}$	7.48	0.46
$P_{D,3}$	-7.17	0.67
$P_{D,4}$	3.68	0.46
$P_{D,5}$	-0.81	0.13
$P_{N,1}$	$2.1 \cdot 10^{-4}$	$7.7 \cdot 10^{-4}$

Table 3.2: Standard deviations, σ, of the estimated ARX parameters.



(a) Original output compared to the estimated output using 5th order ARX and estimated output using 5th order ARX with first $\tau = 5$ output entries set to zero.

(b) Prediction error plot of the estimated ARX model.

Figure 3.10: Results of the ARX model estimation with $(n_{P_D}, n_{P_N}, \tau) = (5, 1, 5)$.

model:

$$y(k) - \hat{y}(k) = \frac{1}{P_D(q)} e(k).$$

One of our system identification objectives, see the beginning of this chapter, was that the model prediction error $y(k) - \hat{y}(k)$ has to be colored, i.e., $|y(1) - \hat{y}(1)| \ll |y(30) - \hat{y}(30)|$. So we can assume that the weight of $\frac{1}{P_D(q)}$ on the equation error does not have enough freedom to get the write prediction error structure. Thus we try another parametric model structure with different weights on the equation error, like ARMAX or PEM [11]. The results are again poor and for that reason are not presented in this report. The reason why will become intuitively clear once we have shown the obstacles in ARX parameters estimation from IR data.

At the beginning of this section we made an assumption that the equation error $e(t)$ is white noise. It is possible that the chosen parametric model structure is not useful for the identification of our test system from impulse response data. In Subsection 3.3.1 we give some explanation why the ARX model gives such a poor result. We also will see why it is not surprising that the parameter $P_{N,1}$ has such a big variance.

We try to improve the ARX model estimate by setting the first few values of our output sequence equal to zero. For our test system these are the first 5 values of the IR. We are allowed to do this because we have already assumed that the first 5 seconds are the dead-time of our system, i.e., $\rho(t) = 0$ for $t \in [0, 5]$. Next, we use exactly the same ARX structure (3.10) but a slightly modified IR to fit a model to our test system.

The resulting IR is shown in Figure 3.10(a) (circles line). Compared to (3.11) this gives a slightly better estimate for our system. But this fit is still very poor. In Subsection 3.3.1 we give some reasons why the parametric model structure gives such poor results.

3.3.1 Why ARX model does not work?

In the previous section we have seen that the system identification results from IR using the ARX model structure are very poor. Here we provide an intuitive explanation for this problem. For this purpose want to have a closer look at the following points:

- the available experimental data (pulse input and impulse response output)

- the chosen model structure (ARX)
- the identification method (least squares)

and the combination of the three. As already said at the beginning of Section 3.3, from the practical point of view it is not often the case that we have to identify the system from its measured IR. Possibly because of this reason there is not much theory on system identification using IR. Thus at first it seemed reasonable to try a simple ARX model structure to identify our system. In this subsection we give some reasons why the ARX model structure is insufficient for system identification from IR.

An ARX model gives us a parametric fit to the given (experimental) input-output data of the system. In our case the input (in discrete time) has only one non-zero value, because it is a discrete impulse input: $u(k) = (1 \;\; 0 \;\; \dots \;\; 0)^{\mathrm{T}}$.

Let us now have a closer look at the estimation procedure of the ARX model parameters. One of the biggest assumptions for the ARX model structure, next to the linearity and time-invariance of the system, is that the equation error $e(k)$ is white noise (zero expectation, finite variance and uncorrelated). The parameters of the ARX model are computed by minimizing the quadratic equation error [1], $e(k)$, the so-called Least Squares Estimation (LSE):

$$\min_{P_D,P_N} \|e(k,P_D,P_N)\|^2 = \min_{P_D,P_N} \|P_D(q)y(k) - P_N(q)u(k)\|^2. \tag{3.12}$$

To solve the LSE problem of Eq. (3.12) we set up the parameter equations in a matrix form (for clarity the vector with equation errors on the right side of the equation is left out):

$$\begin{pmatrix} y_{n+1} \\ y_{n+2} \\ \vdots \\ \vdots \\ y_N \end{pmatrix} = \underbrace{\left(\begin{array}{cccc|ccc} y_n & y_{n-1} & \dots & y_1 & u_{n+1-\tau} & \dots & u_{n+1-\tau-m} \\ y_{n+1} & y_n & \dots & y_2 & u_{n+2-\tau} & \dots & u_{n+2-\tau-m} \\ \vdots & \ddots & & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ y_{N-1} & y_{N-2} & \dots & y_{N-n} & u_{N-\tau} & \dots & u_{N-\tau-m} \end{array}\right)}_{F} \left(\begin{array}{c} a_1 \\ a_2 \\ \vdots \\ a_n \\ \hline b_0 \\ \vdots \\ b_m \end{array}\right), \tag{3.13}$$

[1] In the literature the equation error is sometimes called the estimation error.

where N is the number of data points, n and $m+1$ are the orders of $P_D(q)$ and $P_N(q)$ polynomials, respectively, τ is the time-delay. In our case the input vector has only one (the first one) non-zero entry, $u(k) = (1 \;\; 0 \;\; \ldots \;\; 0)^{\mathrm{T}}$. Thus the matrix F in Eq. (3.13) reduces to $\tilde{F}$:

$$\tilde{F} = \left(\begin{array}{cccc|cccc} y_n & y_{n-1} & \cdots & y_1 & 0 & \ldots & 0 & u_1 \\ y_{n+1} & y_n & \cdots & y_2 & \vdots & & \vdots & 0 \\ \vdots & \ddots & & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & & & \vdots \\ y_{N-1} & y_{N-2} & \cdots & y_{N-n} & 0 & \ldots & 0 & 0 \end{array}\right), \tag{3.14}$$

under the condition that $n = \tau + m$, otherwise all entries in the right part of matrix $\tilde{F}$ in (3.14) would be zero.

Putting $\tilde{F}$ back in (3.13), we see that we can estimate only the parameter b_m. This is why with the given IR data we can estimate only one parameter in the nominator of the transfer function $P_N(q)$ (see Results in Section 3.3).

To make the matter even worse, to estimate the only parameter of $P_N(q)$ we have just one equation, see (3.13) and (3.14):

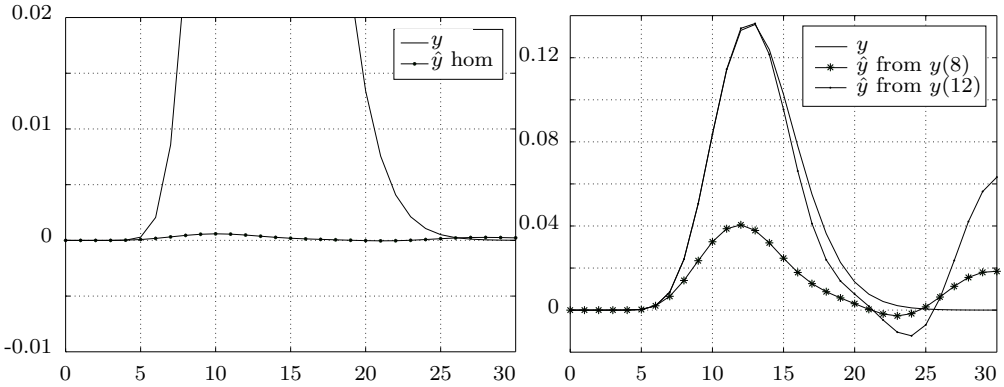
$$y_{n+1} = a_1 y_n + \ldots + a_n y_1 + b_m u_1. \tag{3.15}$$

We examine (3.15) using the results from Section 3.3. There we fitted an ARX model of orders $n = 5$, $m + 1 = 1$ and $\tau = 5$. The best results we got by setting the first $\tau = 5$ values of the output equal to zero. In that case (3.15) reduces to $y_6 = b_0 u_1$ and the b_0 parameter is defined at once by the first non-zero output value $y_{\tau+1} = y_6$. From the IR of our test system we know that this output value is very small. Intuitively, we can see that with a larger value of $y(t)$, $t > 6$, we get a different value for the parameter b_0. The results show that the standard deviation of b_0 is indeed large.

We have explained why the combination of the available data and the model structure give poor estimation results, for the given test system.

Another guess is that the homogeneous part of the system is non-negligible. One usually assumes that because of stability of the system the homogenous part can be neglected. We estimate the homogenous part of the system by fitting an AutoRegressive (AR) structure to the output (the same structure as ARX in (3.10), but with $u(k) = 0, \forall k$). The results of fitting an AR structure of order 5 are presented in Figure 3.11.

From Figure 3.11(a) we see that the homogenous part is negligible, as we would normally have expected. Here the first five output values are taken



(a) Output and the estimated homogenous part of the system.

(b) Homogenous part estimate of 5th order with different initial conditions.

Figure 3.11: Estimated homogenous part of the system.

as the initial state of the system. Evaluating the AR structure with even more values of output $y(k)$ as the initial conditions gives us better results, see Figure 3.11(b). In any case, as we have seen in other sections of this chapter, we can approximate our system with a 5-th order model. But a 5-th order AR model does not give good results, therefore we exclude the possibility that the homogenous part of the system is non-negligible.

The result of yet another modification is presented in Figure 3.12. We take the 'best' result of Section 3.3. The best ARX model was found by putting the first $\tau = 5$ entries of the output equal to zero. The numerator of the estimated transfer function is very small, see the dotted line in Figure 3.12. We scale the estimated output by the original output. The estimated ratio is our new parameter in the numerator $P_{N,1}$. In Figure 3.12 the resulting IR is presented as dashed line. We see that roughly the first 10 seconds give a pretty good estimate. The new parameter is a kind of amplification of the 'old' estimate. Of course the whole IR is amplified and that is why after 25 seconds we see a big deviation of the new estimate from the original output.

We have given some reasons why the parametric structure ARX gives poor estimation results of our test system. Nevertheless, we did not give a proof that the ARX structure is wrong for our input-output data. In [12] the reason why ARX does not work for the impulse input is stated in a formal way for a simple first-order ARX model structure. In essence, the impulse input is not exciting enough and that results in an inconsistent estimator for $P_N(q)$: the estimate $\hat{P}_N$ in mathematical terms does not converge to the true

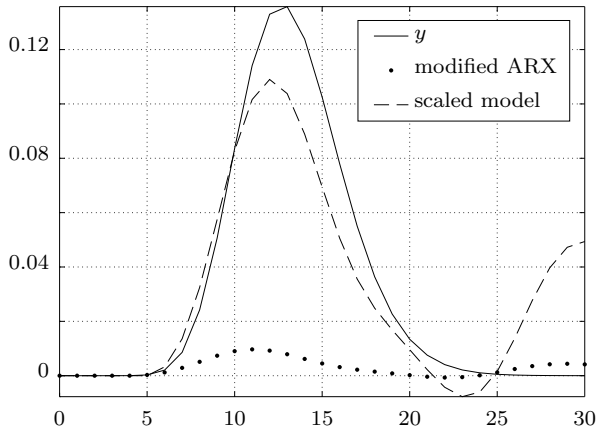


Figure 3.12: The IR of the 5th order ARX model fit with first $\tau = 5$ values set to zero and the IR of the modified ARX model with a scaled parameter $P_{N,1}$.

value of $P_N(q)$ independent of the size of the experimental data. In [12] this is shown for a simple case of first-order ARX model, that the estimator $\hat{P}_{N,1}$ is not consistent, i.e., it does not converge to the true value of $P_{N_0,1}$.

3.4 Maximum likelihood method

Background

Until now we did not account for the positivity characteristic of our system while performing system identification. In [5] an overview is given of the problems in minimal positive system identification. It states that for high-order systems the order of positive realization is often larger than the order of the minimal system realization without the positivity constraints. This shows that it is a difficult problem to find a minimal positive realization of a system.

In this section we are going to use a different, compared to the previous sections, approach for the identification of a system. Namely, the likelihood function maximization [2, 11]. We use a result from [4], where a likelihood function is maximized subjected to the positivity aspect of the system.

The key idea of this system identification method is to assume some probability distribution for the (experimental) data. We find the parameters

of the supposed distribution by maximizing the joint probability for the whole data-set.

We make the following assumptions about our system:

- The output of the system follows a Poisson distribution.

  This is justified when the bacteria coming out of the reactor follow a discrete Poisson process, where the number of bacteria in nonoverlapping time intervals is independent for all intervals.

- The system is positive and linear.

  This assumptions we already made before when we derived the mathematical model of the system, see Section 2.1.1.

- The system is of third order.

  As we have seen from the previous sections, we can estimate the test RTD function with a third order system.

So our problem as described in [4] is to find the residues and the eigenvalues of a continuous-time single-input-single-output positive LTI system with the following IR $h(t)$:

$$h(t) = r_1 \exp(\lambda_1 t) + r_2 \exp(\lambda_2 t) + r_3 \exp(\lambda_3 t), \tag{3.16}$$

subject to the following conditions:

(1) $r_1 > 0$,

(2) $r_1 + r_2 + r_3 \geq 0$,

(3) $(\lambda_1 + \bar{\delta})r_1 + (\lambda_2 + \bar{\delta})r_2 + (\lambda_1 + \bar{\delta})r_3 \geq 0$,

(4) $(\lambda_1 + \delta)^2 r_1 + (\lambda_2 + \delta)^2 r_2 + (\lambda_3 + \delta)^2 r_3 \geq 0$,
$\forall \delta \quad \text{s.t. } -\lambda_3 \leq \delta \leq \bar{\delta} \quad \text{with} \quad \bar{\delta} := -\dfrac{\lambda_1 + \lambda_2 + \lambda_3 - 2\sqrt{\Theta}}{3}$, where
$\Theta := (\lambda_2 - \lambda_3)^2 + (\lambda_1 - \lambda_2)(\lambda_1 - \lambda_3)$,

(5) $\lambda_3 < \lambda_2 < \lambda_1$,

together with the boundary conditions for all six parameters: $|r_i| \leq R$ and $\lambda_m \leq \lambda_i \leq \lambda_M, i = 1, 2, 3$, where R, λ_m, λ_M are some constants. The above conditions are necessary and sufficient for the existence of a third-order positive system realization from a measured IR, for more details see [4].

The Likelihood function is defined as the joint probability density function of the process variables. In our case it is the product of the discrete Poisson probabilities that at time k the experimentally measured concentration of bacteria z_k is equal to an integer value n_k, for $k \geq 1$. Thus the Likelihood function we want to maximize is the following:

$$L(\theta) = \prod_{k=1}^{N} \mathbb{P}(z_k = n_k) = \frac{\alpha_1^{n_1} \cdot \ldots \cdot \alpha_N^{n_N}}{n_1! \cdot \ldots \cdot n_N!} \exp\left(-\sum_{k=1}^{N} \alpha_k\right), \tag{3.17}$$

where N is the number of measurements of our system and α_k is the intensity coefficient of the bacteria output (Poisson) process, which is closely related to the IR $h(t)$ in (3.16). α is defined as follows:

$$\alpha_k(\theta) = \frac{1}{\Delta} \sum_{j=1}^{3} \frac{r_j}{\lambda_j} e^{\lambda_j k \Delta} (1 - e^{-\lambda_j \Delta}),$$

where Δ is a time step. It is a function of the vector of the unknowns $\theta = \begin{pmatrix} r_1 & r_2 & r_3 & \lambda_1 & \lambda_2 & \lambda_3 \end{pmatrix}^{\mathrm{T}}$.

Results

We maximize the Likelihood function (3.17) by using the Matlab optimization algorithm *fmincon*. The direct implementation of this system identification method in Matlab does not work. This is possibly due to numerical problems related to the evaluation of the likelihood function. A possible solution for this problem in Matlab could be providing the Gradient and the Hessian of the Likelihood function and the nonlinear optimization constraints explicitly, which are needed to perform the optimization. In Matlab this could give better estimation results and reduce the number of *fmincon* algorithm iterations. Unfortunately, this adjustment did not work either. In some way or another, the optimization routine of *fmincon* stops after just a few iterations and returns the initial value θ_0 as the optimal solution for our problem.

In Appendix B the Gradient and the Hessian of our likelihood function are given in explicit form. To make the numerical calculations easier we minimize the logarithm of the likelihood function.

3.5 A brief review of the results

In this chapter we used four different methods to estimate a RTD function. Some results did not agree with our expectations, for example a time-domain parametric ARX structure was a poor fit to our test RTD. An intuitive explanation for this is that the pulse input does not excite the system enough to use a parametric model.

We have seen that the ERA from realization theory gave the best identification results. Comparing the parametric ARX model to the ERA method we can see that ERA weights all the output values at once, while ARX 'looks' one step back at every given time. To compare the two methods in formula form, it is easier to consider the state equation: ERA method weights $x(k) = A^{k-1}B$ and the ARX method $x(k) \approx Ax(k-1) \approx A^2x(k-2)$ etc..

In previous section we have tried to estimate our test system by a positive system of third order. Unfortunately, the implementation in Matlab did not give the desired results. Overall, finding a positive realization of a system is known as a difficult problem.

Chapter 4

Controller design

There exist many control design approaches. The choice of a specific design approach depends on the objectives and the goals of the controller. In Chapter 2 we derived a bilinear state-space representation of our system. In [3] these systems are called 'nearly linear', which gives us a good motivation for designing a controller based on the linearized model of our system.

In this chapter we design a controller for the linearized system around an equilibrium point using two different approaches. In Section 4.1 we use classical control rules to design a Proportional Integral Derivative (PID) controller. Next, we improve the performance and robustness of the controller using a lead compensator. In Section 4.2 we use a modern control theory to design a Linear Quadratic Gaussian (LQG) controller. We compare the performance of the controllers by their time constant - the rise time of the system's response to a step input, robustness and implementation difficulty. In Section 4.3 we implement both linear feedback controllers on the nonlinear model of our system.

Design specifications and goals

Before designing a controller for any system we first have to understand the characteristics of the system and define the goals of the controller. As we have seen in Chapter 2, our system has two inputs and one output, for a schematic representation, see Figure 4.1. In the control literature $u(t)$ usually denotes the input to the controller. In our case it denotes the bacteria input into the system, which we cannot control. The controller input in our case is the lamp intensity, which we can adjust up to some limiting value.

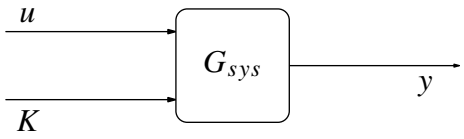


Figure 4.1: Schematic representation of a nonlinear system with two inputs and one output.

When the system is linearized around an equilibrium point, we can 'unlink' the state-space system in two parts: one from the bacteria input to the bacteria output and the other from the lamp intensity input to the bacteria concentration output, see Figure 4.2. Furthermore, in the linearized model both the inputs and the output represent the deviations from the equilibrium values, i.e., $u^*(t) = u(t) - u^{\text{eq}}$, $K^*(t) = K(t) - K^{\text{eq}}$ and $y^*(t) = y(t) - y^{\text{eq}}$. Thus, we have to keep in mind that we do not design a controller for the real inputs $u(t)$ and $K(t)$ and the real output $y(t)$, but rather for the deviations of these quantities from the equilibrium state of the system, $u^*(t)$, $K^*(t)$ and $y^*(t)$, respectively.

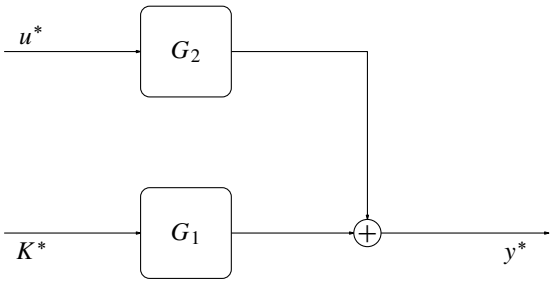


Figure 4.2: Schematic representation of the linearized system.

In this chapter we use feedback control theory to design a suitable controller for our system. For schematic representation of the system with a feedback, see Figure 4.3. Here the plant represents the "controllable" part of the system, which in our case is the subsystem G_1 with the lamp intensity input $K^*(t)$ and the bacteria concentration output $y^*(t)$. Noise in our case is the input of the system that we cannot control. Because we have no prior knowledge of the (statistical) properties of the incoming bacteria, we will assume that $u^*(t)$ is a stochastic noise with some positive non-zero expectation.

The most important objectives for the design of a feedback controller for the linearized system are: stability, robustness and good performance. Stability is the most essential requirement for any system. In mathematical

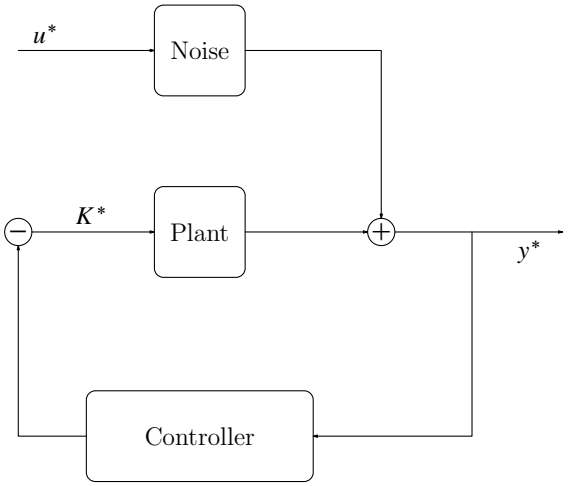


Figure 4.3: Representation of the linearized system with a feedback.

terms it means that we require all poles of the closed-loop transfer function to have negative real part. With a feedback controller it is possible to achieve stability even when the initial plant of the system is unstable. We want the behavior of our system also to be robust against (small) perturbations of the plant and variations of the noise. To achieve good performance of the system the controller has to respond quickly to changes in the input (here by input we mean both lamp intensity and bacteria inputs of the system). Furthermore, we prefer a simple controller above a complicated controller. This is because a simpler controller is easier to implement and it is easier to understand its dynamics.

Equilibrium point of the linearized system

Usually it is not difficult to compute an equilibrium point of the system, i.e., where the state does not change over time: $\dot{x}^{\text{eq}}(t) = 0$. In our project we use a model derived using the RTD function. In Chapter 3 we used system identification methods to estimate the RTD. Using the estimated RTD function in our model gives some complications, as we will see next.

We have to assume some reasonable value for the parameter α of our model, since we have no knowledge about the actual parameters of the system. In Subsection 2.1.2 we defined α as the destruction rate of bacteria by the UV light inside the reactor. Solving the equilibrium equations of (2.14)

by eliminating the state and rearranging the equation, we get:

$$\frac{y^{\mathrm{eq}}}{u^{\mathrm{eq}}} = C(-\alpha K^{\mathrm{eq}} I + A)^{-1}(-B). \tag{4.1}$$

Thus we can calculate a lamp intensity when the system is in an equilibrium for a certain destruction ratio of bacteria $y^{\mathrm{eq}}/u^{\mathrm{eq}}$. In chemical processes literature the term 'log-reduction' of bacteria is usually used to indicate this bacteria destruction coefficient. If we want to achieve a 4-log reduction of bacteria, it means that 99,99% of bacteria have to be destroyed by the UV light inside the reactor.

In Figure 4.4 the relation (4.1) is presented for different RTD estimates of matrices (A, B, C). Namely, order 5, 6 and 20. The parameter α is chosen as $\alpha = 0.55$. The nominal test system (3.5)–(3.6), indicated by its order, $n = 20$, is shown for comparison. On the horizontal axis are the equilibrium values of the lamp intensity and on the vertical axis are the destruction ratios of bacteria. We can see that for bacteria reduction of less than 2-log (less

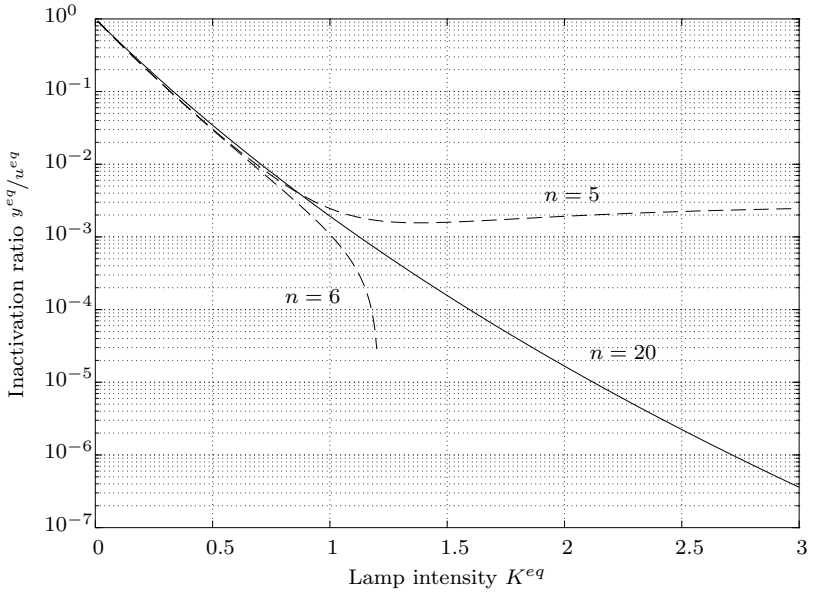


Figure 4.4: Inactivation ratio $y^{\mathrm{eq}}/u^{\mathrm{eq}}$ dependent on K^{eq} for different RTD models of order $n = 5$, $n = 6$ and $n = 20$.

than 99%), we get approximately the same value for the equilibrium lamp intensity. But if we want to get a 4-log reduction of bacteria the values of K^{eq} vary a lot. For example, when we use a RTD estimate of order $n = 6$, the inactivation ratio curve even becomes negative at approximately $K^{\mathrm{eq}} = 1.2$.

For the RTD estimate of order $n = 5$, the inactivation ratio curve is relatively flat for some time after $K^{\text{eq}} = 1$, reaching 4-log reduction for $K^{\text{eq}} \approx 30$. For the nominal system of order $n = 20$, 4-log reduction is achieved at $K^{\text{eq}} = 1.6$. This means that although we concluded in Chapter 3 that a reduced order model with $n = 5$ gives a good approximation to our test RTD function, we cannot use this RTD estimate for the model of our system if we want to achieve high log reduction of the bacteria. Therefore we are forced to use a higher order RTD estimate.

Table 4.1 presents the parameters of an equilibrium point of our system. We use these parameter values in the next two sections to design a controller.

parameter	value
RTD order n	6
α	0.55
u^{eq}	1
y^{eq}	10^{-4}
K^{eq}	1.18

Table 4.1: Equilibrium point parameters of the system.

4.1 PID controller

Background

The Proportional Integral Differential (PID) controller is very common in controller design for chemical processes. **P** is the proportional part of the controller - it produces a signal which is proportional to the error between the output signal and the desired output. **I** is the integral part of the controller - the integrated error is fed back into the system. **D** is the derivative part of the controller - the rate of change of the error is fed back to the controller. Because a PID controller takes care of the error signal in three different ways it often produces the desirable robust controller for the system. For more information about robust control theory and controller design see [2, 15].

PID controller has the following transfer function structure:

$$C_{\text{PID}}(s) = g\left(\frac{T_d s}{\epsilon s + 1} + 1 + \frac{1}{T_i s}\right), \tag{4.2}$$

where parameter g is a proportional gain of the controller, T_d is the derivative time and T_i is the integral time of the controller. The purely derivative part of PID controller, $T_d s$, i.e., $\epsilon = 0$, is not technically realizable. For this reason we make the differentiator realizable by choosing $0 < \epsilon \ll T_d$.

It could be very cumbersome to find PID controller parameters by trial-and-error. A straightforward approach was developed to find the parameter values, the Ziegler-Nichols rules. According to these rules we first connect the P-part of the PID controller to the plant. We increase the gain g_0 of the controller until it becomes unstable and estimate the period T_0 of undamped oscillations which occur in the step-response of the loop gain $G_1(s)C_{\mathrm{P}}(s)$. For a schematic representation of the closed-loop see Figure 4.5. Then, the

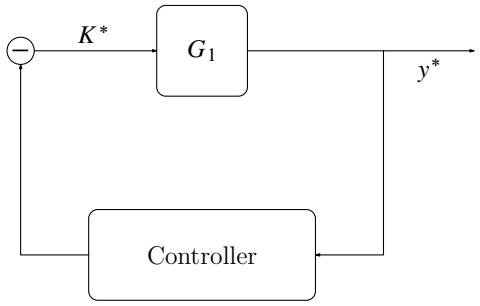


Figure 4.5: Schematic representation of the plant with a feedback controller.

parameters of the PID controller are given by

$$g = 0.6g_0, \quad T_d = 0.125T_0, \quad T_i = 0.5T_0. \tag{4.3}$$

The Ziegler-Nichols rules were developed under the assumption that the transfer function of the plant is of a well damped low-pass type.

We should mention the effect of the three PID actions of the system. First, connecting a P-action to the plant results in a smaller time-constant of the system, i.e., the system responds quicker to the disturbances. Used by itself the proportional action could make the system unstable. To find out which values of proportional gain g make the plant unstable we can look at the root-loci of the plant, which are the trajectories of the roots of the characteristic equation of the closed-loop system $1 + G_1 C_{\mathrm{P}}$. Secondly, the I-action connected to the plant eliminates the steady-state error of the system and makes the time-constant smaller. This comes at the expanse of a large overshoot and a slower response (or settling time) of the system. Thirdly, the derivative action decreases the overshoot of the step-response and makes the system faster at the cost of small steady-state error.

There are different ways of improving a controller. One way to make a system more robust is to use a so-called 'lead compensator'. It creates a phase advance of the system, which is always a good thing because it makes the relative stability margins larger. The lead compensator $C_{\text{lead}}(s)$ has the following transfer function form:

$$C_{\text{lead}}(s) = \frac{\gamma\dfrac{s}{\omega_0} + 1}{\dfrac{s}{\omega_0} + \gamma}, \tag{4.4}$$

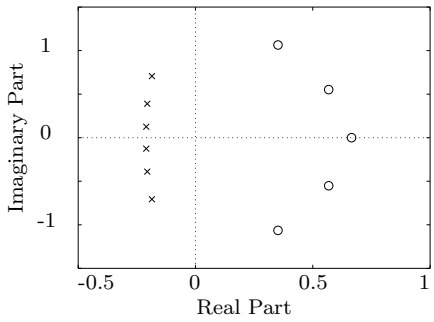
for $\gamma > 1$ and with ω_0 the cross-over frequency around which we want to design a lead compensator. In the frequency interval $(\frac{1}{\omega_0},\ \gamma\omega_0)$ the lead compensator has a differentiating action.

Results

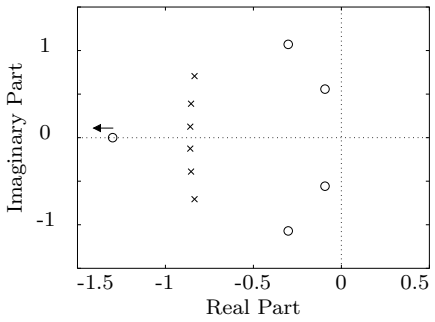
We want to use the Ziegler-Nichols rules to design a robust controller for our system. Before designing a controller we need to understand the system. We look at its stability characteristics and the open-loop step and frequency responses. Because we have two inputs into the system we consider two open-loop subsystems. We can control only the lamp intensity, thus the plant of the system is the transfer function from $K^*(t)$ to $y^*(t)$, which we have called G_1. The goal of this design is to control the bacteria concentration, which is the second input into the system. Therefore, after designing a controller we verify it on the closed-loop system with the transfer function from $u^*(t)$ to $K^*(t)$.

As introduced in the previous section, we call G_1 the plant of our system and $u^*(t)$ the noise of the system. In Figure 4.6 the pole-zero plots of the plant are presented. Figure 4.6(a) shows the poles and zeros of the RTD model of order 6. We can see that all zeros are in the right-half plane, which means that the inverse of this system is unstable. Figure 4.6(b) shows the pole-zero plot of the plant in the linearized system G_1. One of the zeros has a large magnitude as is indicated by an arrow. The former represents the model of the system when the UV lamp is off and the latter represents the linearized model of the system when the lamp is turned on. From the pole-zero plots we can see that just by turning on the lamp results in a stable system and a stable inverse system (all zeros have negative-real part).

The pole-zero plot of the transfer function G_2 from the noise input $u^*(t)$ to the output $y^*(t)$ is shown in Figure 4.7. We can see that this subsystem



(a) Pole-zero plot of the RTD model of order $n = 6$.



(b) Pole-zero plot of the plant of the linearized system.

Figure 4.6: Pole-zero plots.

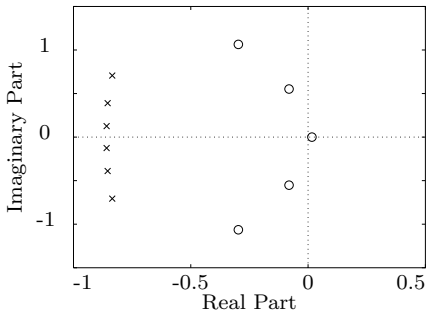
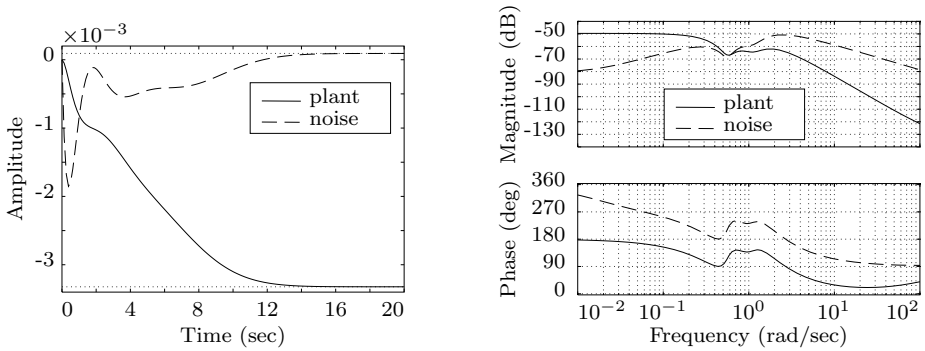


Figure 4.7: Pole-zero plot of the noise subsystem.

is also stable, but has one zero in the right-half complex plane. Notice, that G_1 and G_2 have the same poles.

In Figure 4.8(a) the response in the time-domain of the system to both step inputs is presented. We can see that the plant has a reverse response to



(a) Step responses of the open-loop subsystems.

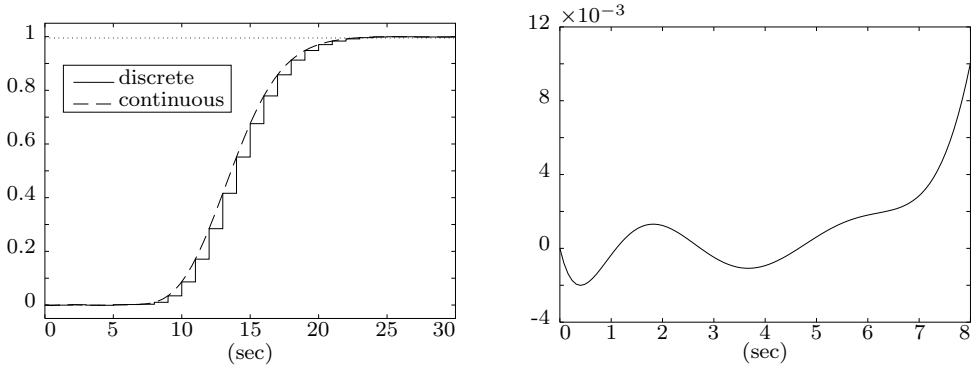
(b) Bode plots of the open-loop subsystems.

Figure 4.8: Step response and frequency response of the system.

a step lamp intensity input. This means that as we turn the lamp intensity higher, the concentration of bacteria will decrease to some new equilibrium. We can see that the noise input converges to the desired bacteria inactivation ratio of 10^{-4}. In the first 2 seconds it looks like the noise subsystem experiences a reverse response. This could be explained by the fact that we use an approximation of the RTD function to model the system. From Figure 4.9 we can see that in the first 5 seconds there is a considerable derivation in the step response of the estimated RTD from the original test RTD function, which is zero for the first 5 seconds. Thus we should keep in mind that our model error is approximately ten times larger than the equilibirum output-value of the bacteria concentration.

Figure 4.8(b) represents the open-loop frequency responses of both subsystems. From the Bode magnitude plot we can see that our plant is low-pass system type. It as well does not have many bumps, suggesting that the system is of low-order. These characteristics of the plant of our system satisfy the assumptions of the Ziegler-Nichols controller design rules.

To be able to use the Ziegler-Nichols rules we have to connect a proportional gain to the plant. In Figure 4.10(a) the root loci of the characteristic polynomial of the closed-loop with proportional gain are presented. It shows how the poles of the closed-loop transfer function behave for different positive values of the controller gain. We can see that there is one pole that goes

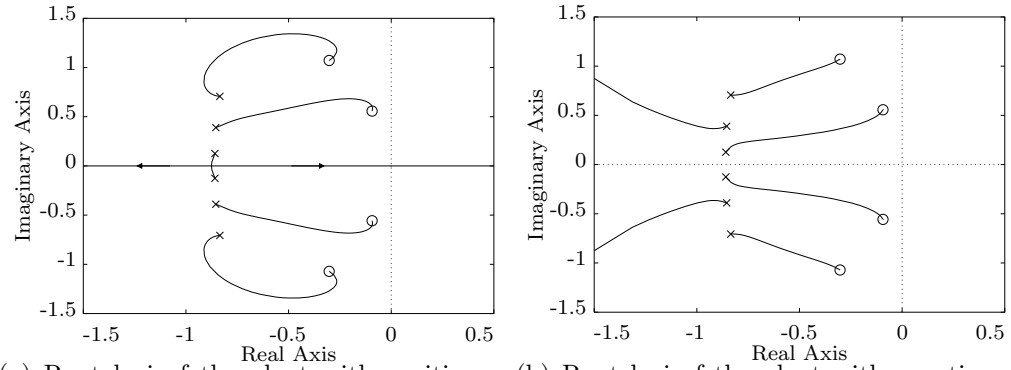


(a) Step Responses of the discrete-time and continuous-time RTD model with $n = 6$.

(b) Zoomed-in step response of the continuous RTD model.

Figure 4.9: Step response of the discrete-time and continuous-time RTD models

to $+\infty$ for gain $g \to \infty$. Usually in control theory the negative value of the



(a) Root loci of the plant with positive proportional gain.

(b) Root loci of the plant with negative proportional gain.

Figure 4.10: Root loci of the plant.

controller is fed back into the system. This rule is based on the assumption that the plant of the system has a positive response to a step input. In our case we have a negative step response, see Figure 4.8(a). Therefore, we have to use a negative gain for the controller.

The root loci of the plant connected to a negative gain are shown in Figure 4.10(b). From the root loci we can see that the system stays stable for all possible (negative) gain values of the feedback. Thus the closed-loop does not become unstable whatever value of the gain we choose and we cannot

apply the Zigeler-Nichols rules. When we connect an integrator feedback $C_{\mathrm{I}}(s) = \dfrac{1}{s}$ to the plant, the closed-loop eventually does become unstable. In Figure 4.11 the root loci of $\dfrac{G_1(s)}{s}$ are presented for gain values $-\infty \leq g \leq 0$.

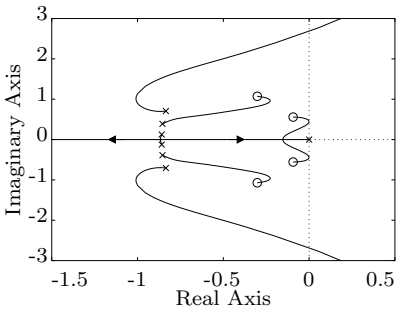


Figure 4.11: Root loci of the plant connected to an integrator.

Now we use the Ziegler-Nichols rules to design a PID feedback controller for the plant with an integrating action, i.e., for $\dfrac{G_1(s)}{s}$. From root-loci plot we derive the smallest value of the proportional gain g_0 for which the system becomes unstable and undamped oscillations occur in the response. In Table 4.2 the values of parameters for a PID controller are presented based on (4.3). As a rule-of-thumb, the parameter ϵ which makes the derivative

parameter	value
g_0	-4377.8
T_0	2.34
g	-2626.7
T_d	0.3
T_i	1.17
ϵ	10^{-3}

Table 4.2: Parameters of the PID controller.

action of the controller realizable, is chosen in the order of $10^{-2}T_d$.

We can validate the designed PID controller in the time- and frequency-domain. As introduced in the beginning of this chapter, we validate the designed controller by the stability, robustness and performance characteristics of the closed-loop transfer function. We call a closed-loop system stable when:

- the transfer functions of the plant and the controller are proper,

- there is no unstable pole-zero cancellation in the closed-loop gain $G_1(s)C(s)$,

- the zeros of the closed-loop characteristic polynomial $1 + G_1(s)C(s)$ have a negative real part.

In our case there is no unstable zero-pole cancellation in the closed-loop gain $G_1(s)C_{\mathrm{I}}(s)C_{\mathrm{PID}}(s)$. Together with stable poles of the closed-loop transfer function, see the pole-zero plot in Figure 4.12, we can conclude that our closed-loop system is stable.

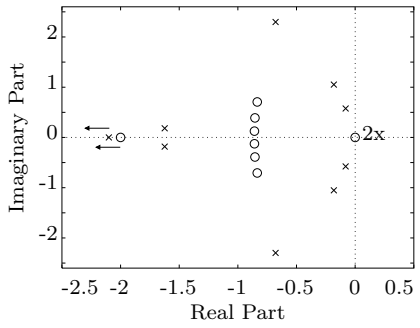
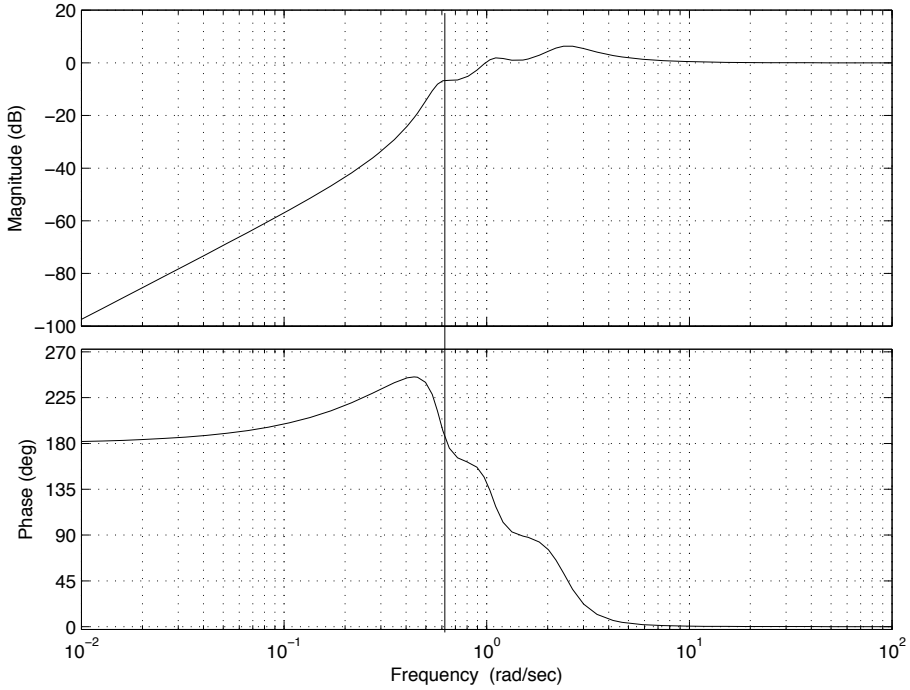


Figure 4.12: The closed-loop pole-zero plot with the I-PID controller.

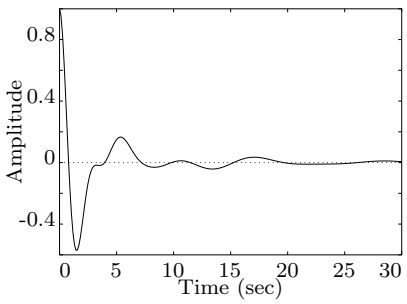
From Figure 4.13(a), we see that the stability margins of our system are very small. This means that the closed-loop system with the I-PID controller can become unstable for small perturbations of the plant or small variations in noise. The vertical line indicates the 'critical' frequency of 180° and the modulus stability margin of the closed-loop system. Because of the poor stability margins, the robustness of this closed-loop system is not sufficient.

Figures 4.13(b) and 4.13(c) present the step responses from both inputs to the output. From Figure 4.13(b) we see that there is a large negative overshoot in the step response in the first few seconds. This indicates that the closed-loop system responds too quickly. Figure 4.13(c) presents the response of the closed-loop system to the step noise input. Again we can see a large overshoot and small oscillations in the first 10 seconds of the response, which are not desirable at all. Thus the performance of the I-PID controller, designed using the Ziegler-Nichols rules, is not sufficient.

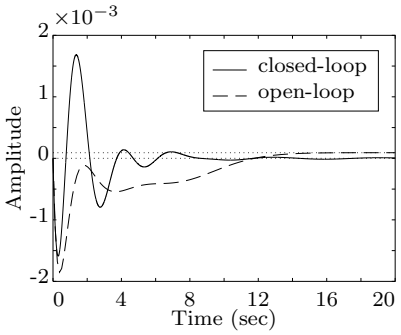
One way to improve the designed controller is to tune the parameter values of the PID controller until we achieve better closed-loop characteristics.



(a) Bode plot of the closed-loop with the I-PID controller.



(b) The closed-loop step response with the I-PID controller.



(c) The closed-loop step response from the noise input to the output.

Figure 4.13: Validation of the PID controller design in time- and frequency-domain.

This could be a tiresome process. Instead, we use a lead compensator to achieve better stability margins of the closed-loop system and thus making the system more robust against (small) perturbations. As a starting point we use again the system with the integrator in the feedback, i.e., the system with loop gain $\frac{g_0 G_1(s)}{s}$ with $g_0 = 1$. By increasing the static gain g_0 we can explore the closed-loop dynamics. Figure 4.14 shows the Nyquist plot of the loop gain with the integral controller of the form $\frac{g_0}{s}$ with $g_0 = 2000$. The 'critical' frequency is the frequency at which the system eventually becomes unstable if we increase the static gain g_0 even further. In Figure 4.14 this frequency region is marked with grey ellipses. For our test system the 'critical' frequency is between 2 and 3 rad/sec.

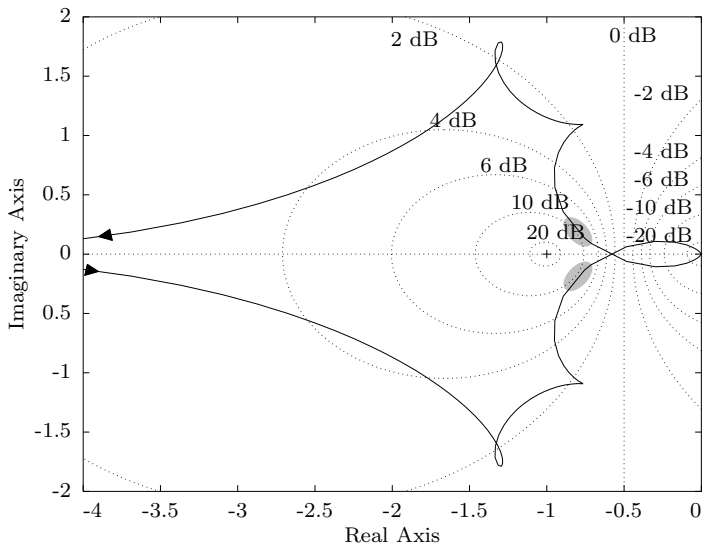


Figure 4.14: Nyquist plot of the loop gain with the integral controller.

We design a lead compensator around frequency $\omega_0 = 2$, to obtain satisfactory stability of the closed-loop system. The Bode plot of the compensator is presented in Figure 4.15. Using this compensator in series with the integrator in the feedback, we achieve phase advance for the loop gain of the system around frequency 2 rad/sec.

Setting the gain equal to the PID-gain g from Table 4.2, we arrive at the following structure for the controller: $\frac{g}{s} C_{\text{lead}}(s)$, which is a compensated integral controller. Again we use the same validation techniques as for the I-PID controller, see Figure 4.16.

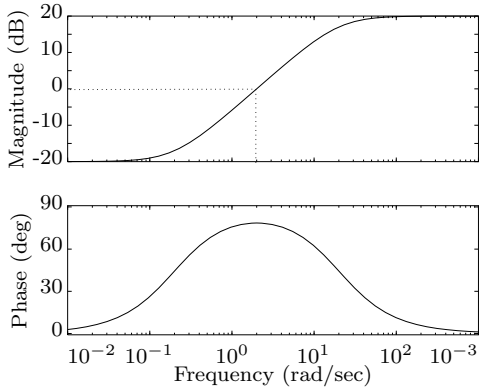
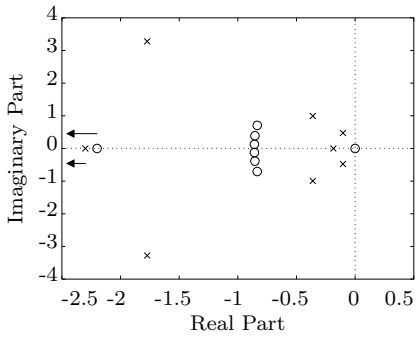


Figure 4.15: Frequency response of the lead compensator with parameters $\gamma = 10$ and $\omega_0 = 2$.

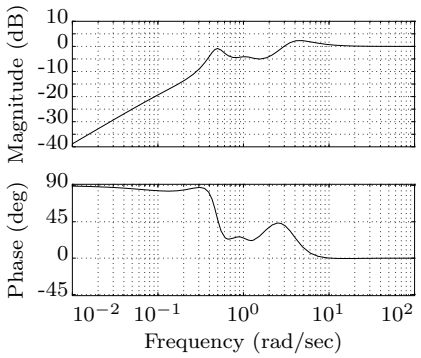
From Figure 4.16(a) we can see that the closed-loop system is stable. The Bode plot of the closed-loop transfer function, in Figure 4.16(b), shows that the system is also robust against perturbations, because the Bode phase is less than 180° even before the Bode magnitude crosses the 0 dB line. The response of the closed-loop system to a step lamp intensity input is presented in Figure 4.16(c). For comparison, the step response of the closed-loop system with the I-PID controller is also shown in the same figure. We can see that the system with lead compensator is slower and has a larger settling time, but the overshoot is reduced considerably. By fine-tuning the lead compensator, which is much easier than fine-tuning a PID-controller, we could get even better closed-loop responses. Due to the lack of time, this is not done in this report.

In Figure 4.16(d) the closed-loop response to a step noise input is compared to the open-loop response to a step noise input. We can see that the closed-loop system is quicker than the open-loop system. The closed-loop system with the compensated integral controller has a much smaller overshoot than the closed-loop system with the I-PID controller, see Figure 4.13(c), as already concluded from the responses of the system to a step lamp intensity input in Figure 4.16(c).

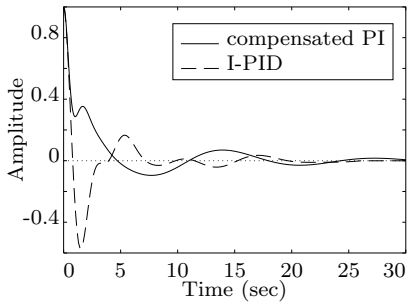
Overall, we prefer the integral controller with lead compensator to the I-PID controller, because on top of all reasons mentioned above, the prior has a simpler structure than the latter.



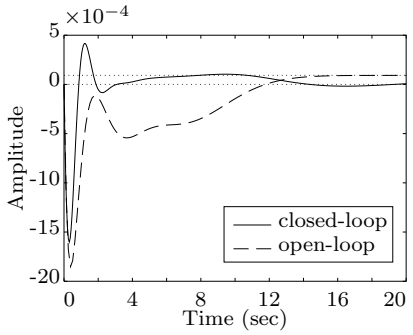
(a) The closed-loop pole-zero pattern.



(b) Bode plot of the closed-loop transfer function.



(c) Step response of the closed-loop system.



(d) Closed-loop step response from the noise input to the output.

Figure 4.16: Validation in time- and frequency-domain of the integral controller with a lead compensator.

4.2 LQG controller

Background

In this section we design a Linear Quadratic Gaussian (LQG) controller for our system [9]. The **L** stands for linear system, **Q** stands for the quadratic criterium that is being minimized and **G** stand for the Gaussian noise of the model. This type of controller is used to achieve optimal disturbance attenuation. It minimizes the following cost function for the control input $K(t)$:

$$J(K) = \mathbb{E}\left(\int_0^\infty Qy(t)^2 + RK(t)^2 dt\right). \tag{4.5}$$

The LQG controller gives a very simple feedback of the form $K(t) = F(t)x(t)$, where $F(t)$ is a Kalman gain and $x(t)$ is the state of the system that is fed back. Because we only know the output of our system, we have to estimate the state.

We use the linearized model (2.16) to design a continuous-time Kalman estimator for the state of the system using the following state-space representation of the plant:

$$\begin{aligned} \dot{x}(t) &= \tilde{A}x(t) + \tilde{B}K(t) + \tilde{G}u(t), \\ y(t) &= \tilde{C}x(t) + v(t), \\ \text{s.t. } \mathbb{E}(u) &= \mathbb{E}(v) = 0,\ \text{var}(u) = U,\ \text{var}(w) = W,\ \text{cov}(u, w) = 0, \end{aligned} \tag{4.6}$$

where $v(t)$ is an added measurement noise, $\tilde{A} = (-\alpha K^{\text{eq}} I + A)$, $\tilde{B} = -\alpha x^{\text{eq}}$, $\tilde{G} = B$ and $\tilde{C} = \frac{1}{\beta}C$, corresponding to the equations (2.16). Both $u(t)$ and $v(t)$ are assumed to be white noise. In Figure 4.17 a schematic representation of Kalman filtering is presented. Thus we use the estimated state $\hat{x}$ together with the optimal feedback gain F as the LQG controller. The state-estimator

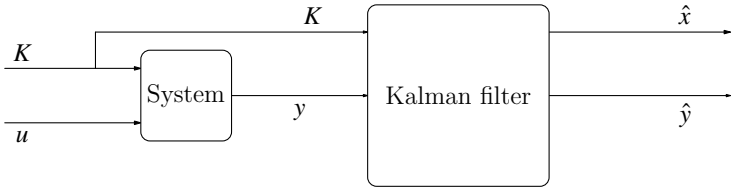


Figure 4.17: Schematic representation of the Kalman filter state estimator.

is constructed by minimizing the steady-state error covariance:

$$\lim_{t\to\infty} \mathbb{E}\left(\|x(t) - \hat{x}(t)\|^2\right)$$

Thus, to design a LQG controller we use the following steps (and functions in Matlab):

- compute a full state feedback (with LQRY),
- compute a Kalman state estimator (with KALMAN),
- connect the state-estimator and the feedback-gain (with LQGREG).

Results

In Figure 4.18 the closed-loop frequency responses of the system using a LQG feedback are presented. The frequency response of the open-loop system is also presented in the same figure, for comparison. The frequency responses are computed from both inputs K and u to the output. We see no difference

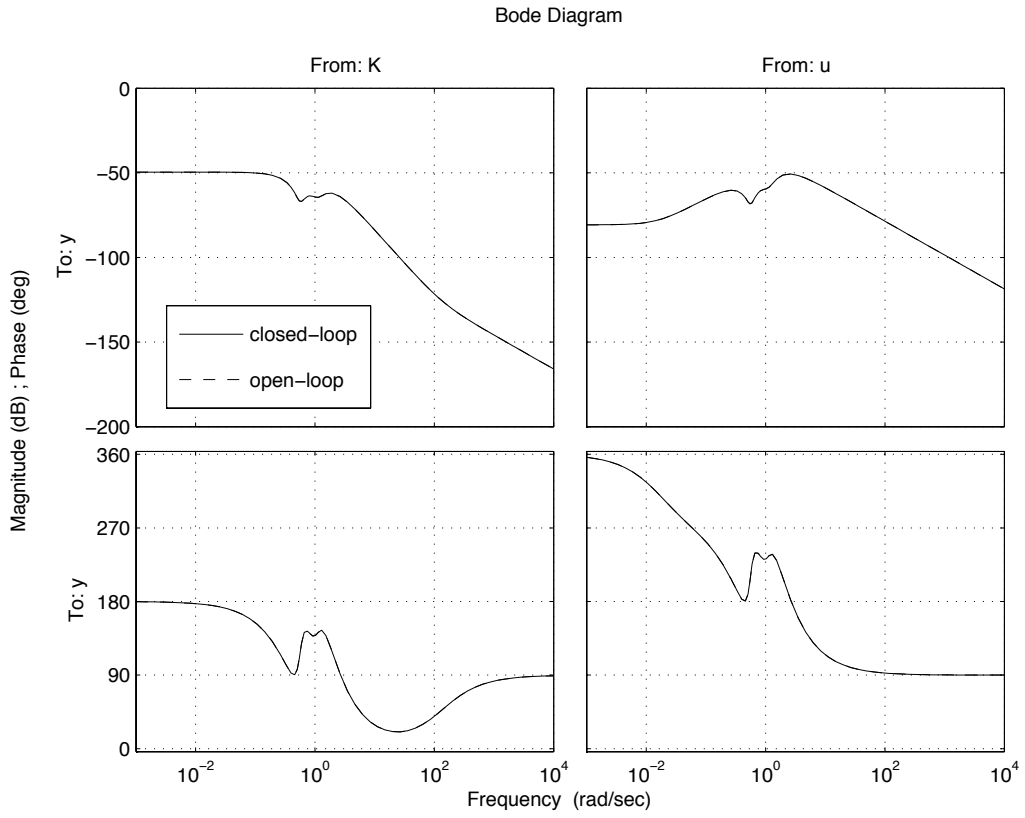


Figure 4.18: Bode plots of the closed-loop with the LQG controller compared to the open-loop responses.

in the responses. At this moment we do not have any idea why the closed-loop response with the LQG controller is similar to the open-loop response of the system.

4.3 Bilinear system with linear feedback

In the previous sections we designed a controller for the linearized model of our system (2.16) using two different approaches. In this section we implement the designed controllers as a linear feedback on the nominal bilinear model of our system (2.13). For a schematic representation of this closed-loop system, see Figure 4.19. Here the bacteria concentration input $u(t)$ consists

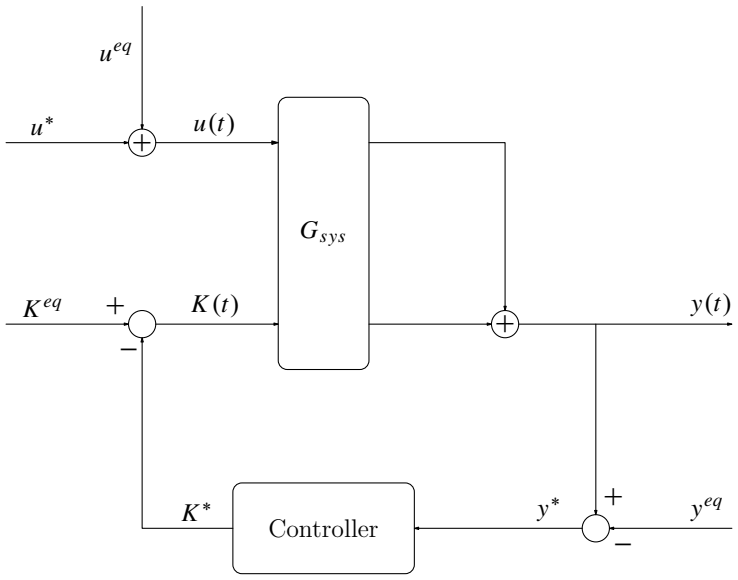


Figure 4.19: Schematic representation of the bilinear system with a linear feedback controller.

of two parts - the equilibrium value u^{eq} and a band-limited white noise term $u^*(t)$. The output of the system $y(t)$ is fed back to the controller by subtracting the equilibrium output value y^{eq} first. The difference between the equilibrium lamp intensity value K^{eq} and the output of the controller $K^*(t)$ is the second input to the nonlinear plant G_{sys}.

We want to compare the performance of the controller designed for the linearized system implemented to the bilinear model of the system. We use Simulink of Matlab to simulate the bilinear system with the three different linear feedback controllers.

First, we implement the integral controller with the lead compensator designed in Section 4.1 in the feedback of the bilinear system. In Figure 4.20 the bacteria input to the system is shown. The bacteria concentration input

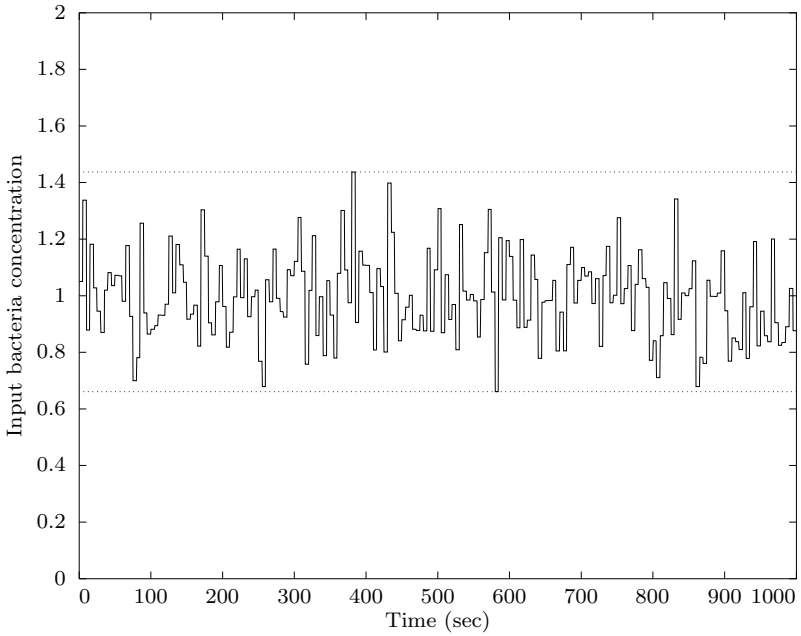
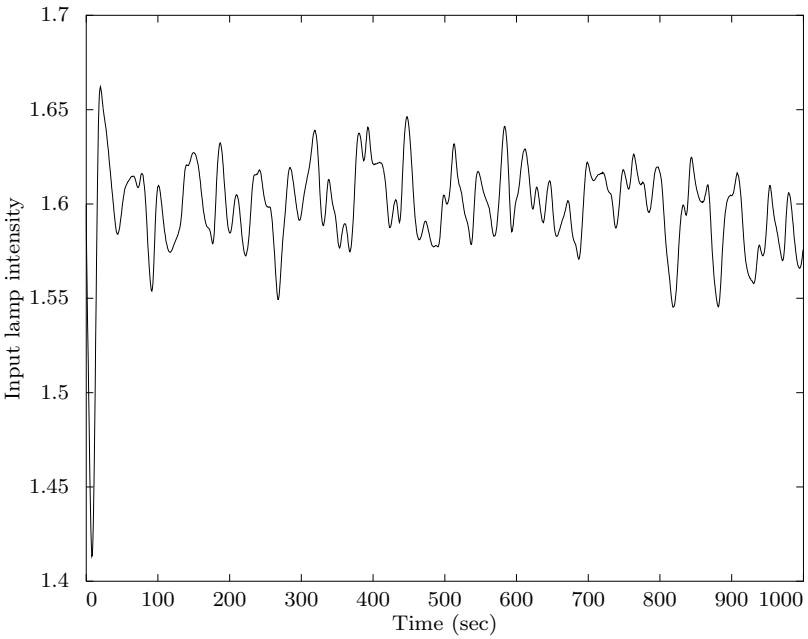


Figure 4.20: Noisy bacteria concentration input into the bilinear system.

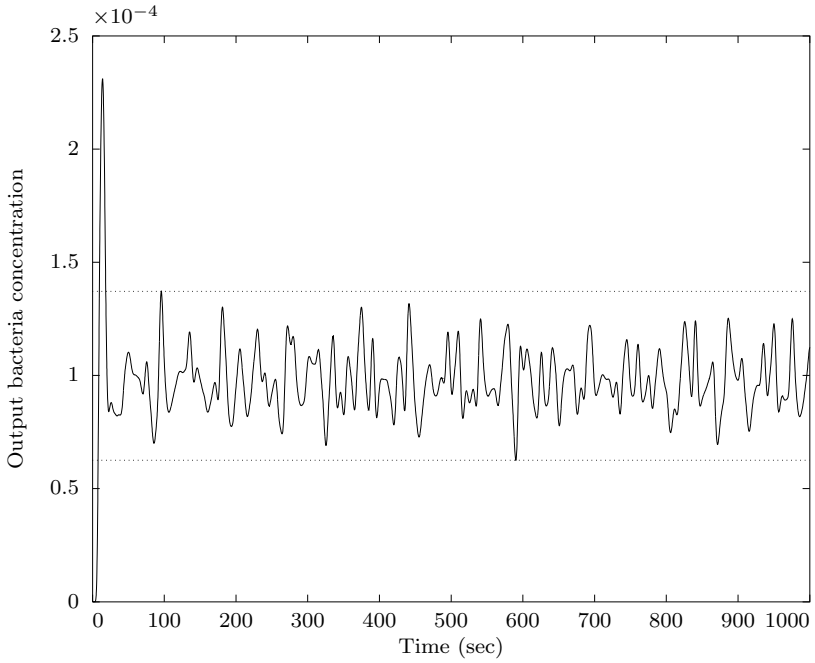
from Figure 4.20 is simulated using a band-limited white noise generator around $u^{\text{eq}} = 1$ and with noise power 0.1 and sample time 5 seconds for $u^*(t)$.

In Figure 4.21 the simulation results of the bilinear system with the linear compensated integral control feedback are presented. In Figure 4.21(a) it is shown how the controlled lamp intensity evaluates in time. Starting at the equilibrium value of $K^{\text{eq}} = 1.6$, the lamp intensity has a negative overshoot in the first 10 seconds of the simulation. After the first 10 seconds it fluctuates around its equilibrium value with less than 5%. The output bacteria concentration is shown in Figure 4.21(b). Apart from a large overshoot in the first 10 seconds, we can be sure that 99,99% of bacteria are inactivated by the UV light inside the system. The overshoot indicated that the designed controller responds too quickly to the derivations in the bacteria on the input of the system.

Next, we implement the LQG controller designed in Section 4.2. Again we simulate the bilinear system with the bacteria concentration input simi-



(a) Lamp intensity input into the bilinear system.



(b) Bacteria concentration output.

Figure 4.21: Simulation results of the bilinear system using the integral controller with lead compensator in the feedback.

lar to the input shown in Figure 4.20. The LQG controller makes very small changes in the lamp intensity, as we can see from Figure 4.22. These small

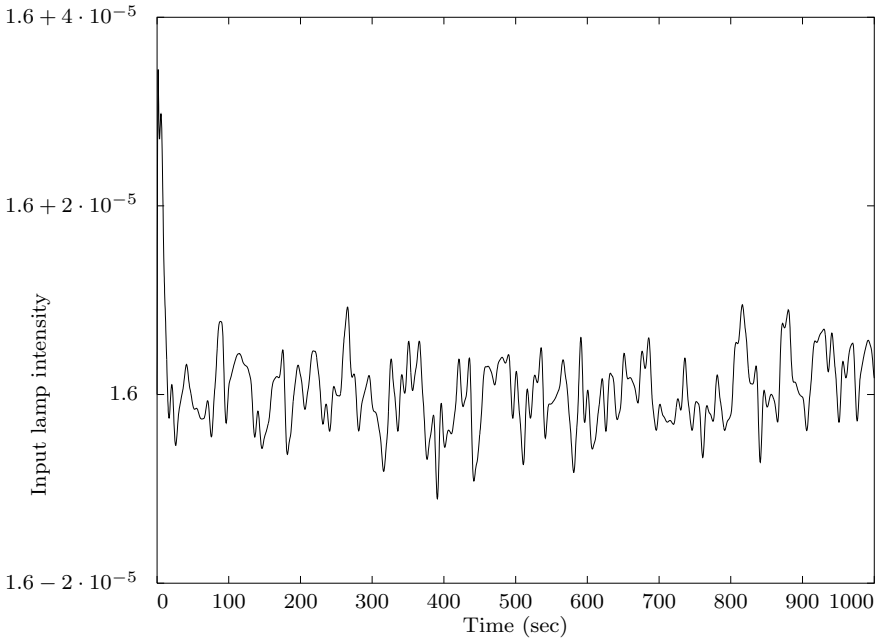


Figure 4.22: Input lamp intensity to the bilinear system with the linear LQG controller feedback.

adjustments of lamp intensity seem sufficient to reduce the bacteria concentration even more than when using the integral controller. See Figure 4.23 for the bacteria concentration output of the simulated bilinear system with the LQG feedback controller. Because in Section 4.2 the closed-loop system with the LQG controller showed surprisingly poor responses, we should be careful with interpreting the results of this simulation.

In the beginning of Section 4.1 we also designed an I-PID controller. The robustness properties of this controller were not sufficient. Nevertheless, we implement this controller in the feedback of the bilinear model of our system. Figure 4.24 shows the simulation results. The lamp intensity curve $K(t)$ shows oscillations with large amplitudes. The bacteria concentration on the output of the system peaks become larger with time. The system is nearly unstable at a certain frequency. Thus, the results are unsatisfactory, as we had already expected.

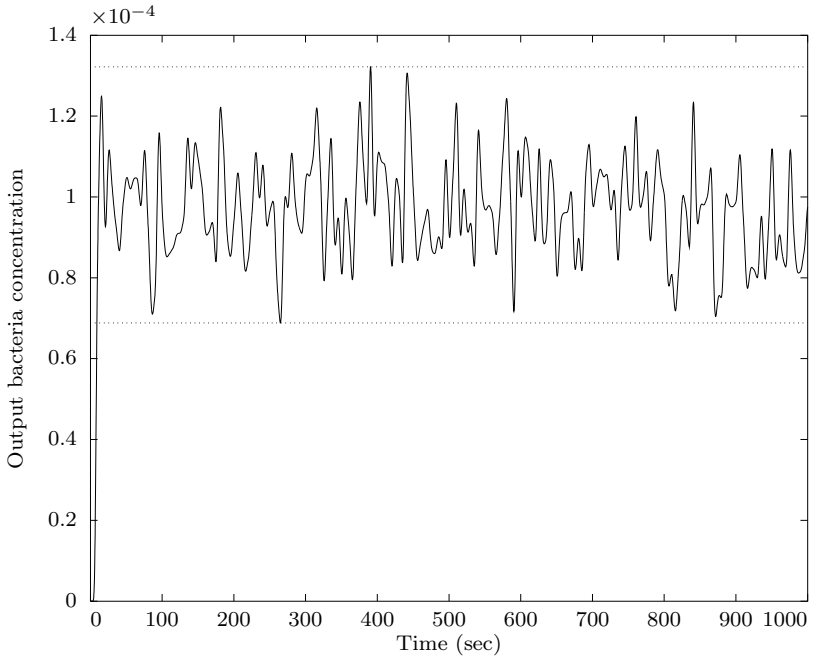
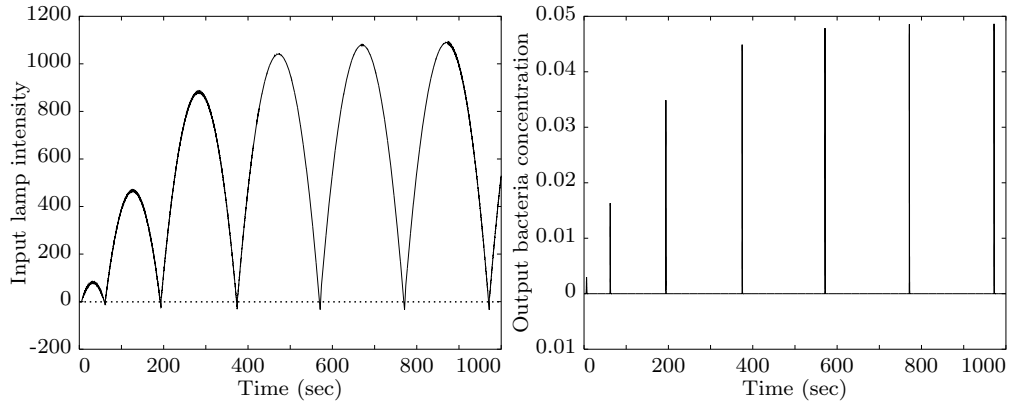


Figure 4.23: Bacteria concentration output of the bilinear system with linear the LQG feedback controller.



(a) Lamp intensity input into the bilinear system.

(b) Bacteria concentration output.

Figure 4.24: Simulation results of the bilinear system using the I-PID controller in the feedback.

Chapter 5

Conclusions and recommendations

In this report we designed a controller for a UV disinfection system, which is on average able to achieve a reduction of 99,99% in the bacteria concentration. This controller is designed for the linearized model of the system. The model is based on the experimentally measured residence time distribution of the bacteria inside the tank. Assuming as well that the reaction of the UV light with the bacteria taking place inside the reactor is of first order, we derived a model for the system which is bilinear in the state.

We compared four different system identification methods to estimate the residence time distribution. The identification was surprisingly not straightforward and the results we got did not always match our expectations. We found out that the identification in time-domain from impulse response is not possible with the parametric model structures, such as ARX. We did get good identification results using methods from realization theory, such as balanced truncation and the eigensystem realization algorithm. In this project we got the best estimation results by using the methods from the realization theory together with an estimated time-delay of the system. In the last section of Chapter 3 we tried, but did not succeed in finding a more realistic *positive* realization of low order of the system. From the literature we know that to find such a positive realization is a difficult task with a lot of not yet solved problems. Nevertheless, for further research it would be interesting to find a (minimal) positive realization of the system.

In the final chapter of this report we designed a controller for the linearized model of our system using two different approaches. The compen-

sated integral controller shows a good performance. When it is implemented on the bilinear model of the system, the integral controller with lead compensation gives us satisfactory results. Implemented on the bilinear model, the designed LQG controller seems to give even better results – without overshoot in the output bacteria concentration. The LQG controller achieves this by making relatively small adjustments of the lamp intensity, which is hard to believe. This would need more research, because at the moment we are unable to explain this result.

In this research we did not use the model with a time-delay to design a controller for the linearized model of the system. Because using a time-delay we got better identification results, we would expect that the controller design for the linearized model with a time-delay to give better performance as well. It is also interesting to see if there are any advantages of designing a nonlinear controller for the bilinear state-space model. Then we could compare the performances of the controller designed for the linearized model to the controller designed for the bilinear model.

Appendices

Appendix A

Mathematical background

A.1 Definitions

Definition A.1.1. A system described by the differential equation $\dot{x} = f(x, u)$ with state $x \in \mathbb{R}^n$ and the input $u \in \mathbb{R}$ is called *positive* if for $x(0) \in \mathbb{R}^n_+$ and $u(t) \in \mathbb{R}_+, \forall t \geqslant 0$, there holds $x(t) \in \mathbb{R}^n_+, \forall t \geqslant 0$, where the positive real numbers are denoted by $\mathbb{R}_+ = \{a \in \mathbb{R}, a \geqslant 0\}$.

Definition A.1.2. The *Hankel Singular Values* are the square root of the eigenvalues of the product of the controllability and observability Gramians, i.e., the square root of the eigenvalues of $\int_0^t e^{A\tau} B B^T e^{A^T\tau} d\tau \cdot \int_0^t e^{A^T\tau} C^T C e^{A\tau} d\tau$.

Definition A.1.3. The matrix $A \in \mathbb{R}^{n\times n}$ is called *Metzler* if all off-diagonal elements of A are nonnegative, $\forall a_{ij} \geq 0$ for $\forall i \neq j$.

A.2 Mathematical norms

In this Appendix we define a few norms used elsewhere in this report.

We denote functions in the time-domain by using lowercase characters, like $g(t)$. We denote functions in the frequency-domain by using uppercase characters, like $G(i\omega)$ with frequency ω. The functions $G(\cdot)$ and $g(\cdot)$ are related by the Laplace transform, $G(s) = \int_0^\infty \exp(-st) g(t) dt$.

We introduce two norms that are closely related to the system characteristics in time- and frequency-domains:

The L_1-norm

$$\|g(t)\|_{L_1} = \int_{-\infty}^{\infty} |g(t)| dt,$$

is the same as the area under the IR plot of the system,

and the H_∞-norm

$$\|G(i\omega)\|_{H_\infty} = \sup_\omega |G(i\omega)|,$$

which is the peak value of the Bode magnitude plot of the system.

The following inequality gives a relationship between the H_∞- and the L_1-norms:

$$\|G(s)\|_\infty := \sup_{s\in\mathbb{C}_0^+} |G(s)| \leq \int_{-\infty}^{\infty} |g(t)|dt =: \|g(t)\|_1. \tag{A.1}$$

From Eq. (A.1) we see that the peak value on Bode plot is smaller or equal to the area under the IR function of a system.

The next relationship also holds:

$$\|g(t)\|_\infty \leq \frac{1}{2\pi}\|G(s)\|_1,$$

where $\|g(t)\|_\infty := \max_{t\in\mathbb{R}} |g(t)|$ is the maximum value of the IR and $\|G(s)\|_1 := \int_{-\infty}^{\infty} |G(i\omega)|d\omega$ is the area under the Bode magnitude plot. To prove this, use the inverse Fourier transform $g(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} G(i\omega)\exp(i\omega t)d\omega$. Thus

$$\begin{aligned}
\|g(t)\|_\infty &= \max_{t\in\mathbb{R}} \left| \frac{1}{2\pi}\int_{-\infty}^{\infty} G(i\omega)\exp(i\omega t)d\omega \right| \\
&\leq \frac{1}{2\pi}\int_{-\infty}^{\infty} \max_{t\in\mathbb{R}} |G(i\omega)\exp i\omega t|d\omega \\
&= \frac{1}{2\pi}\int_{-\infty}^{\infty} |G(i\omega)|d\omega = \frac{1}{2\pi}\|G(s)\|_1.
\end{aligned}$$

A.3 Discrete to continuous system transformation

When discretizing a continuous system a primal assumption would usually be that the systems input $u(t)$ is constant on each discrete time interval $[kT, (k+1)T]$, i.e., $u(kT) = u(kT+\tau)$ for $\tau \in [0, T)$, as in Fig. A.1.

Consider the continuous-time system with state-space representation

$$\begin{aligned}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t)
\end{aligned} \tag{A.2}$$

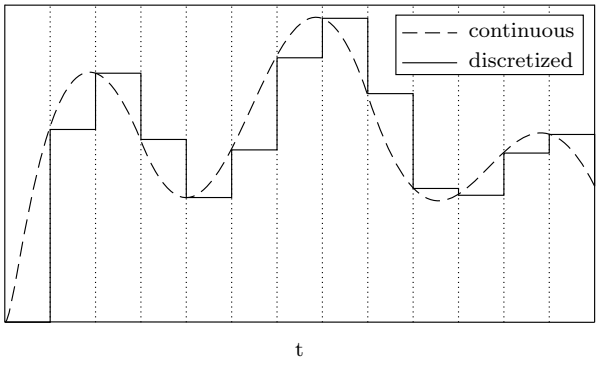


Figure A.1: A discretized continuous signal, the so called Zero-order hold transform of a signal from continuous to discrete time.

and the discrete-time system representation

$$\begin{aligned} x_{k+1} &= A_d x_k + B_d u_k \\ y_k &= C_d x_k, \end{aligned} \tag{A.3}$$

where $t = kT$ and T is the discrete time step. We show that the following relations hold:

$$\begin{aligned} A_d &= \exp(AT) \\ B_d &= \left(\int_0^T \exp(A\tau) d\tau \right) B = A^{-1}(A_d - I)B \\ C_d &= C. \end{aligned} \tag{A.4}$$

The solution of the continuous-time state equation (A.2) is:

$$x(t) = \exp(At)x_0 + \int_0^t \exp(A(t-\tau))Bu(\tau)d\tau$$

and of the discrete-time state equation (A.3):

$$x_k = A_d^{k-k_0} x_0 + \sum_{j=k_0}^{k-1} A_d^{k-1-j} B_d u(j).$$

Comparing the solutions with zero initial conditions, i.e., $k_0 = 0$ and $t_0 = k_0 T = 0$, after one time step T. In continuous time we find

$$\begin{aligned} x(kT+T) &= \exp(AT)x(kT) + \int_{kT}^{kT+T} \exp(A(kT+T-\tau))Bu(\tau)d\tau \\ &\overset{(*)}{=} \exp(AT)x(kT) + \left(\int_0^T \exp(A\tau)d\tau \right) Bu(kT), \end{aligned}$$

where $(*)$ holds because of the Zero-order hold assumption, i.e., the input $u(t)$ is constant on the interval $[kT, kT+T)$, see Figure A.1. In discrete time the solution of the state equation after one discrete time step at time $k+1$ is:

$$x_{k+1} = A_d x_k + B_d u_k.$$

From this we directly see that the relations in (A.4) hold.

A.4 Convolution and Impulse Response

In a continuous-time system the input function $u(t)$ is transformed into the output function $y(t)$. We call this transformation operator $T_t\{u\}$ (the subscript t refers to the transformation in time-domain). We can write down the relation between the input u and the output y of our system as:

$$y(t) = T_t\{u\}.$$

In the special case when the input is a Dirac delta function, $u(t) = \delta(t)$, we call the output of the system the Impulse Response (IR) function, $h(t)$:

$$h(t) = T_t\{\delta\}.$$

We know that we can write any continuous function $u(t)$ in terms of weighted delta functions, by definition:

$$u(t) = \int_{-\infty}^{\infty} u(\tau)\delta(t-\tau)d\tau.$$

For a continuous LTI system, the transformation operator $T_t\{\cdot\}$ must satisfy equalities ① for linearity and continuity and ② for time-invariance. Thus

$$\begin{aligned} y(t) = T_t\{u\} &= T_t\Big\{ \int_{-\infty}^{\infty} u(\tau)\delta(t-\tau)d\tau \Big\} \\ &\overset{①}{=} \int_{-\infty}^{\infty} u(\tau) T_t\{\delta(t-\tau)\} d\tau \\ &\overset{②}{=} \int_{-\infty}^{\infty} u(\tau) h(t-\tau) d\tau \\ &= u(t) * h(t), \end{aligned} \tag{A.5}$$

where the '$*$'-sign denotes the convolution between two functions. Thus in time-domain we can express the output, $y(t)$, of any continuous LTI system as a convolution of the input function, $u(t)$, and the IR of the system, $h(t)$.

Finally, using the causality assumption, $h(t) = 0$, $\forall\ t < 0$, together with (A.5), we get the relation between input $u(t)$ and output $y(t)$ that is presented in Equation (2.1)

$$y(t) = \int_{-\infty}^{\infty} u(\tau)h(t-\tau)d\tau = \int_{-\infty}^{t} h(t-\tau)u(\tau)d\tau,$$

where in (2.1) instead of the IR $h(t)$ there is the RTD function $\rho(t)$.

A.5 Proof Conservation of mass

In the beginning of Chapter 3, see the system identification objectives (B-2.), we claim that the following holds for a causal LTI system with input $u(t) = 0$, $\forall t < 0$,

Claim A.5.1.

$$\int_0^{\infty} h(t)dt = 1 \quad \Longleftrightarrow \quad \int_0^{\infty} u(t)dt = \int_0^{\infty} y(t)dt.$$

Proof. The output can be written in terms of the IR $h(t)$ of the system and the input:

$$y(t) = \int_0^{t} h(t-\tau)u(\tau)d\tau,\ \ t > 0.$$

Substituting this in the equation for the total output and than changing the order of integration, we get:

$$\int_0^{\infty} y(t)dt = \int_0^{\infty}\int_0^{t} h(t-\tau)u(\tau)d\tau dt = \int_0^{\infty}\left[\int_{\tau}^{\infty} h(t-\tau)dt\right]\ u(\tau)d\tau. \tag{A.6}$$

Performing the change of variables $t - \tau = \tilde{t}$ and using our assumption about the total IR $\int_0^{\infty} h(t)dt = 1$, we get:

$$\int_0^{\infty} y(t)dt = \int_0^{\infty} \underbrace{\left[\int_0^{\infty} h(\tilde{t})d\tilde{t}\right]}_{1}\ u(\tau)d\tau = \int_0^{\infty} u(t)dt.$$

Thus we have proven "⇒".

From (A.6) we see directly that $\int_0^{\infty} u(t)dt = \int_0^{\infty} y(t)dt$ is true only if $\int_0^{\infty} h(t)dt = 1$. Thus we have also proven "⇐".

□

Appendix B

Likelihood function derivatives

The Likelihood function in Section 3.4 is defined as:

$$L(\theta) = \prod_{k=1}^{N} \mathbb{P}(z_k = n_k) = \frac{\alpha_1^{n_1} \cdot \ldots \cdot \alpha_N^{n_N}}{n_1! \cdot \ldots \cdot n_N!} \exp\left(-\sum_{k=1}^{N} \alpha_k\right), \tag{B.1}$$

where N is the number of measurements of our system and α_k is a function of the vector of the unknowns $\theta = \begin{pmatrix} r_1 & r_2 & r_3 & \lambda_1 & \lambda_2 & \lambda_3 \end{pmatrix}^{\mathrm{T}}$, defined as follows:

$$\alpha_k(\theta) = \frac{1}{\Delta} \sum_{j=1}^{3} \frac{r_j}{\lambda_j} e^{\lambda_j k \Delta} (1 - e^{-\lambda_j \Delta}),$$

where Δ is a time step.

In Section 3.4 we try to maximize the Likelihood function (B.1) by using the Matlab optimization function *fmincon*. The direct implementation of this system identification method in Matlab does not work. A possible solution for this problem in Matlab could be to provide explicitly the Gradient and Hessian of the Likelihood function and the nonlinear optimization constraints. This could give better estimation results and reduce the number of *fmincon* algorithm iterations.

Next, we calculate analytically the Gradient and Hessian of the Likelihood function. The derivatives of the nonlinear constraints are calculated with Maple and are not shown here.

The Gradient of the Likelihood function

The partial derivatives of $L(\theta) := \prod_{k=1}^{N} L_k(n_k, \alpha_k(\theta))$ with vector of the unknowns $\theta = (\theta)_j$ for $k = 1 \ldots N$ and $j = 1 \ldots 6$ are of the form:

$$\begin{aligned}\frac{\partial L_k(n_k, \alpha_k(\theta))}{\partial \theta_j} &= \frac{\alpha_k(\theta)^{n_k - 1}}{(n_k - 1)!} e^{-\alpha_k(\theta)} \frac{\partial \alpha_k}{\partial \theta_j} - L_k(n_k, \alpha_k) \frac{\partial \alpha_k}{\partial \theta_j} \\ &= \left[L_k(n_k - 1, \alpha_k) - L_k(n_k, \alpha_k)\right] \frac{\partial \alpha_k}{\partial \theta_j}\end{aligned}$$

Here there are two possible derivatives of α_k. First the derivative w.r.t. r_j, i.e., $\theta_1, \theta_2, \theta_3$:

$$\frac{\partial \alpha_k}{\partial \theta_{j=1,2,3}} = \frac{1}{\Delta} \frac{e^{\lambda_j k \Delta}}{\lambda_j} (1 - e^{-\lambda_j \Delta})$$

and the others w.r.t. λ_{j-3}, i.e., $\theta_4, \theta_5, \theta_6$:

$$\frac{\partial \alpha_k}{\partial \theta_{j=4,5,6}} = \frac{1}{\Delta} \left(-\frac{\alpha_{k,j-3}}{\lambda_{j-3}} + k\Delta\alpha_{k,j-3} + \frac{r_{j-3}}{\lambda_{j-3}} \Delta e^{\lambda_{j-3}(k-1)\Delta} \right),$$

where $\alpha_{k,p} = \frac{r_p}{\lambda_p} e^{\lambda_p k \Delta} (1 - e^{-\lambda_p \Delta})$ with in this case $p = j - 3$.

In Matlab there is a function **poisspdf** which computes L_k for a given (array) of n_k's and α_k's.

The derivative of the whole Likelihood function $L(\theta)$ is a little lengthy:

$$\begin{aligned}\frac{\partial L(\theta)}{\partial \theta_j} &= \frac{\partial \left(\prod_{k=1}^{N} L_k(n_k, \alpha_k(\theta)) \right)}{\partial \theta_j} \\ &= \frac{\partial L_1}{\partial \theta_j} \prod_{\substack{k=2 \\ k \neq 1}}^{N} L_k + \frac{\partial L_2}{\partial \theta_j} \prod_{\substack{k=1 \\ k \neq 1}}^{N} L_k + \ldots + \frac{\partial L_N}{\partial \theta_j} \prod_{\substack{k=1 \\ k \neq N}}^{N-1} L_k \\ &= \sum_{i=1}^{N} \frac{\partial L_i}{\partial \theta_j} \prod_{k=1, k \neq i}^{N} L_k\end{aligned}$$

If we take **Log** of the likelihood function, the derivatives seem to become simpler. The Log-likelihood function then becomes $\bar{L}(\theta)$:

$$\overline{L}(\theta) := \log L(\theta) = \sum_{k=1}^{N} \overline{L_k}(n_k, \alpha_k(\theta)) = \sum_{k=1}^{N} \log \frac{\alpha_k^{n_k}}{n_k!} - \sum_{k=1}^{N} \alpha_k$$

So the partial derivatives are:

$$\begin{aligned}\frac{\partial \overline{L_k}(\theta)}{\partial \theta_j} &= \frac{1}{\frac{\alpha_k^{n_k}}{n_k!}} \frac{\alpha_k^{n_k-1}}{(n_k-1)!} \frac{\partial \alpha_k}{\partial \theta_j} - \frac{\partial \alpha_k}{\partial \theta_j} \\ &= \left(\frac{n_k}{\alpha_k} - 1\right) \frac{\partial \alpha_k}{\partial \theta_j}\end{aligned}$$

And the derivative of the whole $\overline{L}(\theta)$ function is, for $\theta = 1, .., 6$:

$$\frac{\partial \overline{L}(\theta)}{\partial \theta_j} = \frac{\partial \left(\sum_{k=1}^{N} \overline{L_k}(\theta)\right)}{\partial \theta_j} = \sum_{k=1}^{N} \frac{\partial \overline{L_k}}{\partial \theta_j}.$$

The **Gradient** is a vector of the partial derivatives:

$$\nabla \overline{L} = \left(\begin{array}{cccccc} \frac{\partial \overline{L}(\theta)}{\partial \theta_1} & \frac{\partial \overline{L}(\theta)}{\partial \theta_2} & \frac{\partial \overline{L}(\theta)}{\partial \theta_3} & \frac{\partial \overline{L}(\theta)}{\partial \theta_4} & \frac{\partial \overline{L}(\theta)}{\partial \theta_5} & \frac{\partial \overline{L}(\theta)}{\partial \theta_6} \end{array} \right)^T.$$

The Hessian of the Log-Likelihood function

The second derivative, the **Hessian**, of the log-likelihood function $\bar{L}$ is a sum of all partial second order derivatives:

$$\frac{\partial^2 \overline{L}}{\partial \theta_i \partial \theta_j} = \sum_{k=1}^{N} \frac{\partial}{\partial \theta_i} \left(\frac{\partial \overline{L_k}}{\partial \theta_j} \right).$$

The second order partial derivative of $\overline{L_k}$ is:

$$\begin{aligned}\frac{\partial}{\partial \theta_i} \left(\frac{\partial \overline{L_k}}{\partial \theta_j} \right) &= \frac{\partial}{\partial \theta_i} \left[\left(\frac{n_k}{\alpha_k(\theta)} - 1 \right) \frac{\partial \alpha_k(\theta)}{\partial \theta_j} \right] \\ &= -\frac{n_k}{\alpha_k^2(\theta)} \frac{\partial \alpha_k(\theta)}{\partial \theta_i} \frac{\partial \alpha_k(\theta)}{\partial \theta_j} + \left(\frac{n_k}{\alpha_k(\theta)} - 1 \right) \frac{\partial^2 \alpha_k(\theta)}{\partial \theta_i \partial \theta_j}\end{aligned}$$

The only unknown in the equation above is the second partial derivative of $\alpha(\theta)$. The Hessian-matrix of $\alpha(\theta)$ is:

$$\nabla^2 \alpha(\theta) = \frac{\partial^2 \alpha_k(\theta)}{\partial \theta_i \partial \theta_j} = \nabla^2_{(i,j)} = \left(\begin{array}{cccccc} 0 & 0 & 0 & \nabla^2_{(1,4)} & 0 & 0 \\ 0 & 0 & 0 & 0 & \nabla^2_{(2,5)} & 0 \\ 0 & 0 & 0 & 0 & 0 & \nabla^2_{(3,6)} \\ \nabla^2_{(4,1)} & 0 & 0 & \nabla^2_{(4,4)} & 0 & 0 \\ 0 & \nabla^2_{(5,2)} & 0 & 0 & \nabla^2_{(5,5)} & 0 \\ 0 & 0 & \nabla^2_{(6,3)} & 0 & 0 & \nabla^2_{(6,6)} \end{array} \right)$$

The Hessian-matrix is symmetric around the diagonal, so the terms on the lower and upper diagonals are equal ($\nabla^2_{(1,4)} = \nabla^2_{(4,1)}$, so on.) So we only have to calculate 6 second-order partial derivatives.

For $i = 1, 2, 3; j = 4, 5, 6$ and $i = j - 3$ holds:

$$\nabla^2_{\substack{i=1,2,3 \\ j=4,5,6 \\ i=j-3}} = \frac{\partial^2 \alpha_k}{\partial\theta_i \partial\theta_j} = \frac{1}{\Delta}\left(-\frac{\alpha_{k,i}}{\lambda_i r_i} + \frac{k\Delta}{r_i}\alpha_{k,i} + \frac{\Delta}{\lambda_i}e^{\lambda_i(k-1)\Delta}\right),$$

and for the partial derivatives on the diagonal of the Hessian of $\alpha(\theta)$, i.e., for $i = j = 4, 5, 6$ holds:

$$\begin{aligned}\nabla^2_{\substack{i=4,5,6 \\ i=j}} =& \frac{\partial^2 \alpha_k}{\partial\theta_i \partial\theta_j} \\ =& \frac{1}{\Delta}\left(\left[\frac{2}{\lambda^2_{i-3}} - \frac{2k\Delta}{\lambda_{i-3}} + (k\Delta)^2\right]\alpha_{k,i-3}\right. \\ &\left. + \left[-\frac{\Delta+1}{\lambda_{i-3}} + k\Delta^2 + (k-1)\Delta\right]\frac{r_{i-3}}{\lambda_{i-3}}e^{\lambda_{i-3}(k-1)\Delta}\right).\end{aligned}$$

The Hessian of $\overline{L}$ is then a 6×6 matrix of the sums of the second partial derivatives $\dfrac{\partial^2 \overline{L}}{\partial\theta_i \partial\theta_j}$.

Bibliography

[1] A. Berman, M. Neumann, and R.J. Stern. *Nonnegative matrices in dynamic systems*. A Wiley-Interscience publication, 1989.

[2] O.H. Bosgra, H. Kwakernaak, and G. Meinsma. Robust control, 2007.

[3] C. Bruni, G. DiPillo, and G. Koch. Bilinear systems: an appealing class of "nearly linear" systems in theory and applications. *IEEE Transactions on automatic control*, AC-19(4), 1974.

[4] A. De Santis and L. Farina. Identification of positive linear systems with poisson output transformation. *Automatica*, 38(5):861–868, 2002.

[5] B. De Schutter. Minimal state-space realization in linear system theory: an overview. *Journal of computational and applied mathematics*, 121:331–354, 2000.

[6] H. S. Fogler. *Elements of chemical reaction engineering*. Prentice-Hall, New Jersey, 3rd edition, 1999.

[7] J. N. Juang and R. S. Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *Journal of Guidance, Control and Dynamics*, 8(5), 1985.

[8] T. N. Koutchma, L. J. Forney, and C. I. Moraru. *Ultraviolet light in food technology:Principles and Applications*. Taylor and Francis Group LLC, 2009.

[9] H. Kwakernaak and R. Sivan. *Linear optimal control systems*. John Wiley and Sons, Inc., 1972.

[10] L. Ljung. *System identification toolbox. For use with MATLAB*.

[11] L. Ljung. *System identification: theory for the user*. Prentice-Hall, New Jersey, 1987.

[12] T. Söderström and P. Stoica. *System identification*. Prentice Hall International (UK), 1989.

[13] Lenntech Water treatment and purification Holding B. V. Uv disinfection [web], 1998-2009. http://www.lenntech.com/systems/uv/uv-disinfection.htm.

[14] S. Van Mourik. *Modelling and control of systems with flow*. PhD thesis, University of Twente, 2008.

[15] K. Zhou. *Essentials of robust control*. Prentice-Hall, New Jersey, 1998.