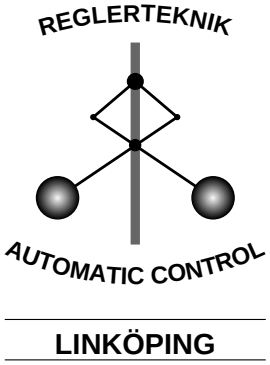
Linköping Studies in Science and Technology
Thesis No. 948

Identification, Diagnosis, and Control of a Flexible Robot Arm

Måns Östring



Division of Automatic Control
Department of Electrical Engineering
Linköpings universitet, SE–581 83 Linköping, Sweden
WWW: http://www.control.isy.liu.se
E-mail: mans@isy.liu.se

Linköping 2002

Identification, Diagnosis, and Control of a Flexible Robot Arm

Department of Electrical Engineering,
Linköpings universitet,
SE–581 83 Linköping,
Sweden.

ISBN 91-7373-356-3
ISSN 0280-7971
LiU-TEK-LIC-2002:21

Printed by UniTryck, Linköping, Sweden 2002

To .... (fill in your name)

Abstract

The most important factors in manufacturing are quality, cost, and productivity. The trend is towards lighter robots with increased mechanical flexibilities, and therefore there is a need to include the flexibilities in the robot models to obtain good performance of the robot.

The core theme in this thesis is modeling and identification of the physical parameters of an ABB IRB 1400 industrial robot. The approximation made is that the robot arm can be described using a finite number of masses connected by springs and dampers. It has been found that a three-mass model gives a reasonably good description of the robot when moving around axis one. The physical parameters of this model are identified using off-line and on-line algorithms. The algorithms are based on prediction error methods. For the off-line identification the MATLAB™ System Identification Toolbox is used. For the on-line identification the algorithm used is a modified version of a recursive prediction error method to cope with continuous time models.

The models are then used in diagnosis and control. Two ways of doing diagnosis using on-line identification are investigated. Estimating some of the physical parameters of the robot arm recursively makes it possible to monitor important aspects of the system such as friction and load.

LQG control of the flexible robot arm is also studied with the aim of good disturbance rejection. Aspects that have been studied are unstable regulators and the use of accelerometers.

Acknowledgments

First of all I would like to thank my supervisor Professor Svante Gunnarsson, without whom I would never have finished this thesis and Dr. Mikael Norrlöf for being my co-supervisor.

Then I would like to express my gratitude to Professor Lennart Ljung for drafting me to the group, and, of course, to the whole group for creating the atmosphere that makes it "endurable to work when the research is going slow". I especially thank Ulla, our secretary, for all the help, and for being the glue that holds the whole group together.

This work was supported by VINNOVA's center of excellence ISIS (Information Systems for Industrial Control and Supervision), which is gratefully acknowledged. The work in this thesis would not have been possible without the support from ABB. I would especially like to thank "my industrial supervisor" Dr. Torgny Brogårdh and also Stig Moberg. Thank you for all the fruitful discussions when I have been visiting you in Västerås.

Of my colleges I would especially like to thank Johan Löfberg for all the discussions and questions you have to put up with. Then also Anna Hagenblad and Fredrik Tjärnström that also have been "terrorized by my questions and/or opinions". They have also given valuable help from reading early versions of parts of the thesis.

Extra credit goes also to the computer support group for keeping the computers happy and for all the interesting discussions, and also to Erik Frisk for answering my most troublesome LaTeX questions.

Finally, I would like to thank my family for all the help and support.

Contents

1 Introduction

During the past decades industrial robots have become a very important factor in the manufacturing industry. Robots are applied to new areas each day. To be able to go into new markets the robots often require better performance or lower price. In order to meet these demands the physical robot structures are built lighter and weaker. Therefore the demands on the accuracy of the robot models, used in the controllers, are growing. Good models are also needed for model based diagnosis of robots. The development rate of new industrial robots is high. This means that there is a need for an automated way of building mathematical models, and also of estimating the parameters in the mathematical models.

To motivate diagnosis of industrial robots two visions for the future can be:

> The first vision includes a diagnosis system for each robot in a production system. There is also a superior diagnosis system that communicates with each robot. This system decides optimal times for maintenance stops in the plant. For a car manufacturing line with several hundred robots there is a large amount of money to be saved by doing plant stops for maintenance only when needed, and also knowing which parts that should be replaced.
>
> The second vision is that the robot gives an alarm when, for example, the

operator has given incorrect load parameters. The robot then suggests that the parameters should be reestimated, and if the operator agrees the load parameters are estimated, and the performance of the robot is improved.

This thesis studies a part of the visions using flexible arm models with a focus on identification, diagnosis and control. The cornerstone is the building of physical models, which can be used for control and diagnosis, and particularly the estimation of the physical parameter values in the models using both off-line estimates and on-line estimates.

A fundamental property in robot control and diagnosis is that the amount of sensors is limited. Usually the only variable that is measured is the position of the motors of the robot arm. This means that a model is needed to compensate for the flexibilities in the robot arm, and to estimate the position of the tool. This is done via mathematical models. If the tool is changed, or any other physical aspect of the arm is changed, and the properties of the model are not changed accordingly, the mathematical model is not valid any more. As a result of this the estimate of the arm position is becoming worse, and the performance of the robot will deteriorate. To prevent this from happening this thesis also deals with how to detect and isolate when some physical aspects of the robot arm are changed.

1.1 Thesis outline

The thesis is organized as follows:

Chapter 2 gives an introduction to the robotics area in general, and ABB robots and the robot used in this thesis in particular.

In Chapter 3 the modeling of a flexible robot arm is studied. Models representing different approximations of a flexible arm are used. The robot arm has a gear box, which has a flexibility. This suggest that the flexible arm may be approximated with two masses and a spring and damper pair. In further investigations a three mass model is used with good results.

In Chapter 4 a short introduction to system identification using prediction error methods is given.

The identification, using prediction error methods, of the robot arm from real data is shown in Chapter 5, both for physical models and for black-box models. The off-line identification is utilized in recursive identification and in diagnosis (Chapters 6 and 7) to estimate nominal parameter values.

A recursive counterpart of the identification is given in Chapter 6.

Diagnosis of the robot is studied in Chapter 7. Three chosen faults, that can be explained by parameter changes, are studied using the algorithms and models from the previous chapters.

In Chapter 8 LQG control for obtaining good disturbance rejection properties is studied. The model used in the design of the regulator can, for example, be acquired using off-line identification (Chapter 5).

Finally in Chapter 9 some conclusions and pointers to future work are discussed.

1.2 Contributions

The main contributions of the thesis are the following:

- The procedure to derive the user defined state space model for system identification using the graphical modeling environment MathModelica and Mathematica in Chapter 3.
- The off-line physical parameter identification applied to an industrial robot in Chapter 5.
- The recursive counterpart of the physical parameter identification in Chapter 6.
- The fault diagnosis procedure based on parameter faults in Chapter 7.
- The discussion about the stability and disturbance rejection of a robot arm controller when only the motor position is measured in Chapter 8.

Parts of the results in this thesis have been previously presented at different conferences:

> Östring, M., Gunnarsson, S., and Norrlöf, M. (2001b). Closed loop identification of the physical parameters of an industrial robot. In *Proceedings of the 32nd International Symposium on Robotics, ISR*, Seoul, Korea.
>
> Gunnarsson, S. and Östring, M. (2001). On regulator stability in control of flexible mechanical systems. In *Proceedings of the 32nd International Symposium on Robotics, ISR*, Seoul, Korea.

To appear at conferences:

> Norrlöf, M., Tjärnström, F., Östring, M., and Aberger, M. (2002). Modeling and identification of a mechanical industrial manipulator. In *Proceedings. of the 15th IFAC Congress*, Barcelona, Spain

Östring, M., Tjärnström, F., and Norrlöf, M. (2002b). Modeling of industrial robot for identification, monitoring, and control. In *Proceedings of the International Symposium on Advanced Control of Industrial Processes*, Kumamoto, Japan

Östring, M. and Gunnarsson, S. (2002). Recursive identification of physical parameters in a flexible robot arm. In *Proceedings of the 4th Asian Control Conference, ASCC*, Singapore, Singapore

An extension to the work in Chapter 5 will be published as a journal paper:

Östring, M., Gunnarsson, S., and Norrlöf, M. (2002a). Identification of an industrial robot during one axis movements. *Accepted for publication in Control Engineering Practice*

2

Robotics

What is a robot? One way to answer that question is to describe what robots are used for. They are utilized for automation in a way that humans have specified. They are built to serve people. This picture of the robot as a nice machine serving humans is, however, not shared by everyone. There are several examples of evil robots in literature and movies, like the T800 and the improved T1000 in the Terminator movies. To make robots behave more nicely the writer Isaac Asimov postulated "The three laws of robotics" that the robot should obey (Asimov, 1950).

1. A robot may not injure a human being or, through inaction, allow a human being come to harm.
2. A robot must obey orders given it by human beings except where such orders would conflict with the First Law.
3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

There is no general definition of what a robot is, but there exist typical characteristic features for robots. They should be flexible and programmable in contrast to automated machines which are made for certain tasks, for example, to screw a cap

on a bottle. There is a wide range of robot types spanning from underwater vessels controlled by computer programs to robots trying to mimic humans, so called humanoids.

This thesis focuses on robots with a computer controlled mechanical arm, so called industrial robots. Industrial robots are in general used in a wide variety of applications. These include arc welding, spot welding, material handling, gluing, grinding, polishing, painting, and much more. The broad spectrum of applications puts strong demands on the robots in, for example, programmability and performance. One definition of this type of robots is made by the Robot Institute of America, RIA (Spong and Vidyasagar, 1989).

RIA: "A robot is a reprogrammable multi functional manipulator designed to move material, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks."

One of the origins of this type of robot is given by a patent from 1954 where a 5-axes manipulator for flexible assembly was described, although is was never implemented. Another milestone appeared in 1969 when General Motors ordered 26 robots from Unimation for spot welding (Bolmsjö, 1992). From this time on robots have become a significant part of the manufacturing industry. Another important step was when fully electrical robots were introduced by ASEA (now ABB) in 1973, the IRB 6 (IRB stands for Industrial RoBot). This resulted in more accurate control which made it possible to use robots in many new areas, for example, arc welding.

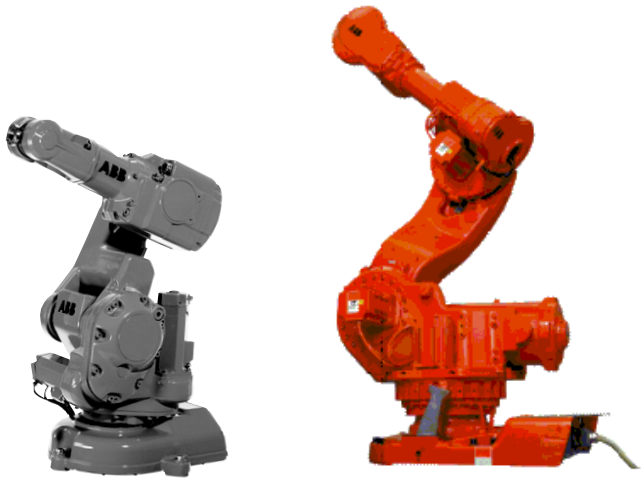
2.1 ABB industrial robot family

In the experiments in this thesis a robot from ABB is used. We will therefore look at the family of robots manufactured by ABB and what parts they contain. In this work the manipulator ABB IRB 1400 is used. To control the manipulator a controller, version S4C, is used (ABB Flexible Automation, 1997a). Apart form the hardware there is also a lot of software involved.

The software in the controller is only a small part of the software used when manufacturing with robots. ABB has a concept of "Industrial IT ... the next way of thinking" (Secher, 2001). It is claimed that this way of thinking reduces the production time from supplier of the parts, manufacturing the goods to the delivery to the customers. This is done by "real-time" information and delivery. Everyone know everything at all time. With this new way of manufacturing it is claimed that everyone will benefit from lower costs and faster delivery. The focus is on software

allowing easy configuration and information exchange. For the robot community this means, among other things, producing general controllers that operate in a "plug and play" fashion.

The robot family of ABB includes small robots, like the IRB 140, which has a height of 109 cm when the arm is aligned vertically. The family spans to the biggest IRB 7600 which has a capacity of lifting 500 kg, and has an installed motor power of 87 kW. The IRB 1400 used in this work is one of the smaller robots in the ABB robot family.



(a) IRB 140 (b) IRB 7600

Figure 2.1 *The smallest and the biggest manipulators from ABB, IRB 140 and IRB 7600.*

2.2 The robot system

The robot system consists of a controller cabinet, a teach pendant and the manipulator. In the cabinet there are driver units for the motors and computers for the control, see Figure 2.2 for a picture of the controller, S4C. The cabinet also contains connections to the environment, like I/O ports and a network interface. The teach pendant is used when programming or manually moving the robot. A

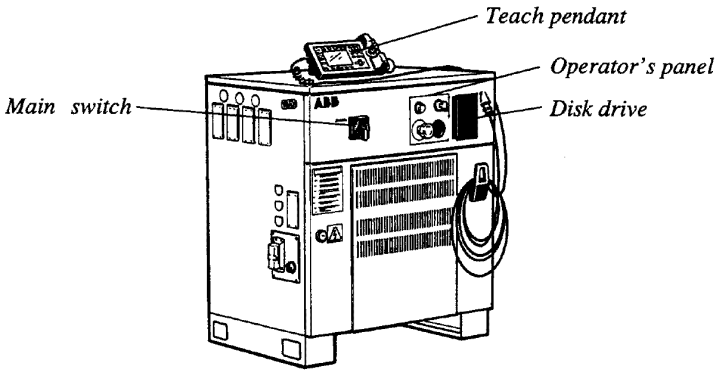


Figure 2.2 *The controller, S4C.*

Figure 2.3 *The teach pendant used for programming or manually moving the robot.*

picture of the teach pendant can be seen in Figure 2.3. The manipulator IRB 1400 is the mechanical arm carrying out the desired tasks. It is shown in Figure 2.4. In addition, an industrial robot application comprises conveyor belts and fences built up to support the robot, and make it safe. In this work we will only study the core of the robot system. We will therefore look at the controller and manipulator more closely in the following sections.

2.3 The manipulator, IRB 1400

The manipulator has six degrees of freedom. The structure and the joints can be seen in Figure 2.4. The size of the manipulator is given by Figure 2.5. The motors

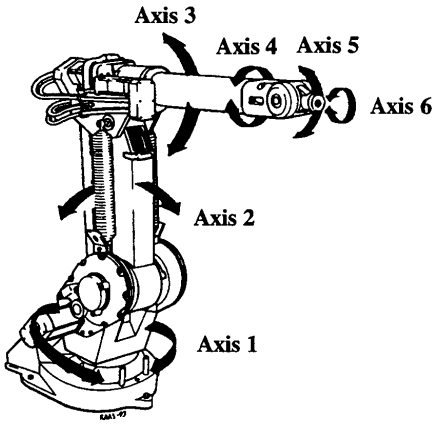


Figure 2.4 *The manipulator IRB 1400.*

of joint one, two and three are placed at the bottom of the manipulator. This is possible because the movement of joint three is controlled by a beam translating the movement at the bottom up to axis three. The motors of joint four, five and six are placed on the back of the upper arm, see Figure 2.4. The motor torque is transmitted via a transmission in the upper arm. The motors are placed in this way to get as much inertia as possible near the center of the robot. Therefore it is possible to have smaller motors compared to if the motors were placed at the actual joints. One needs less torque to give the same acceleration, and one can also make the structure lighter. There are also balancing springs parallel to axis two. These springs are used to balance the robot and decrease the load on the motor for joint two.

The manipulator is also equipped with brakes. The brakes are used when the robot is not running to prevent the robot from falling to the ground. They are also used for emergency stop.

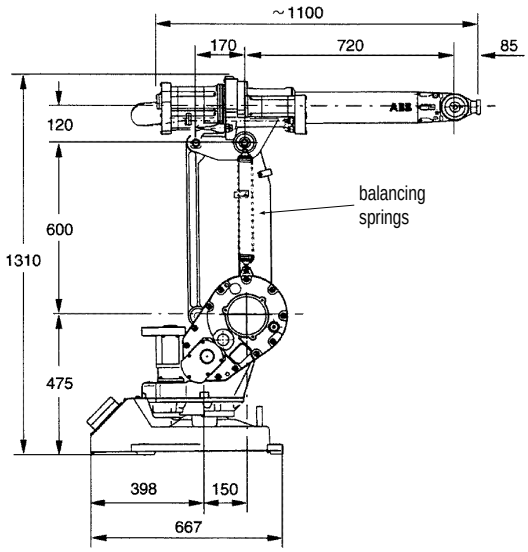


Figure 2.5 *The IRB 1400 manipulator.*

2.4 The controller, S4C

The main parts of the controller are the cabinet with the contained hardware and the teach pendant. In the cabinet the main processor that runs the most part of the software is a Motorola 68060 processor. For the low level control, a DSP (Digital Signal Processor) from Texas Instrument is used. The processors are placed at different boards. There are also other types of boards that can be inserted to increase the functionality. The system used in this work has a memory board with 16Mb of memory, and an optional board for Ethernet communication. Inside the cabinet there is also a possibility to connect digital or analog I/O for controlling tasks in the surroundings. The control system is also prepared for communication with, for example, a PLC (Programmable Logic Controller) through standard bus interface.

The cabinet also contains drive units for the motors. The standard configuration includes drive units for four to six motors depending on the number of degrees of freedom (DOF) of the robot. Degrees Of Freedom is the number of independent motions an object can perform in relation to a coordinate system. In our case this means six. There is also a possibility to connect up to twelve motors controlled

by one cabinet. This is, for example, useful for controlling simpler robots holding the workpiece as the case in many arc welding applications where there is a 2-DOF robot for moving and turning the object.

A disk drive is also a part of the cabinet. It can be used to load or store programs from diskettes.

As said previously the controller, S4C, also includes a teach pendant shown in Figure 2.3. On the teach pendant there is a 3-DOF joystick. It can be used in several modes. In one mode it is possible to position the tool in a Cartesian coordinate system corresponding to the base frame. The base frame is a Cartesian coordinate system fixed to the ground. In another mode the tool can be reoriented. It is also possible to move individual axes of the robot and external axes.

The teach pendant also has a display and a keyboard. They can be used to program the robot in conjunction with moving the robot. The display also shows what the robot is doing, and displays error messages. It can also be used as user interface by an application. For example, it can display a custom made menu where the operator can choose what the robot should do next.

2.5 The controller software

There are a lot of different softwares involved when programming and running the robot. First of all the control system needs to be booted and the programs loaded. This is done using floppy disks or Ethernet. Ethernet is used because of the connection with MATLAB™ and the existing test environment which demands it. When booting the system, the system first connects to a BOOT-server, then files are downloaded via a FTP-server. For further communication and transfer of files an NFS-server is used. Network File System (NFS) is used for sharing files across a network, see (IETF Secretariat, 2001) for more information.

The programming language for the commercial system is called RAPID™ (ABB Flexible Automation, 1997b; Nilsson, 1996). This is used for controlling the robot using simple instructions like:

Example 2.1 Simple RAPID™ example
A RAPID™ instruction can look like

```
moveL p2,v100,fine,tool0
```

which moves the robot from the current position and orientation of the tool, with the speed `v100` to the point and orientation `p2`. The point and orientation

`p2` can be set with the use of the teach pendant. Just move to the position and orient the tool in the desired way with the joystick on the teach pendant.

In Example 2.1 a linear path in Cartesian coordinates is specified. The movements in Cartesian coordinates are converted to movements around each joint. This must be done in consideration with limitations such as acceleration constraints.

In RAPID™ it is also possible to do more complex instructions, for example, loops, control structures, write to the display of the teach pendant or read input from the teach pendant etc..

Another programming language is used in the test environment, which has testing and evaluation functionally built in. This language looks very much like the RAPID™ language, but only the core components for the motion control is included to make it easy to build. For example, components having to do with the teach pendant is excluded. On the other hand it is more flexible and easier to extend. These features are utilized together with MATLAB™ (Matlab, 2001) in the experiments in Chapters 5 and 6.

MATLAB™ is used on a terminal which communicates through Ethernet with the controller, see Figure 2.6. With the use of the test language we can log signals

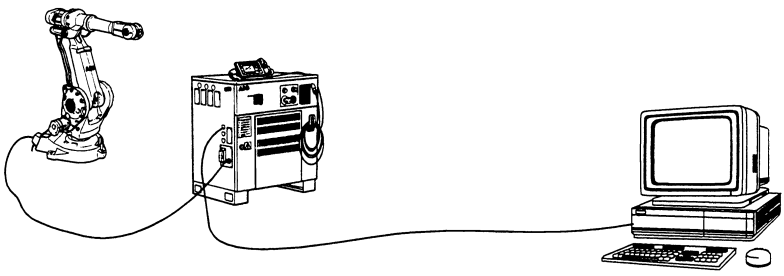


Figure 2.6 *The manipulator connected to the controller which is connected to the terminal (PC) via Ethernet.*

using the "Data-logger". We can also affect signals with the "Data-input" module for doing ILC, see Norrlöf (2000), or system identification as in this thesis.

The interface between MATLAB™ and the controller is mainly built by M. Norrlöf (Norrlöf, 1998). It makes it possible to control and affect signals from MATLAB™. The synchronization between MATLAB™ and the controller is made with semaphores. These semaphores are actual files on an NFS-server where both the robot

and the terminal PC read and write. It is also via this NFS-server the logged data and the applied signals are transfered.

2.6 The control system

Here a typical control structure for controlling robots similar to what is presented in Spong and Vidyasagar (1989) is given. More detailed information about the control blocks can be found in Spong and Vidyasagar (1989). A schematic view of the system is shown by Figure 2.7. It contains the two blocks F_f and F represent-

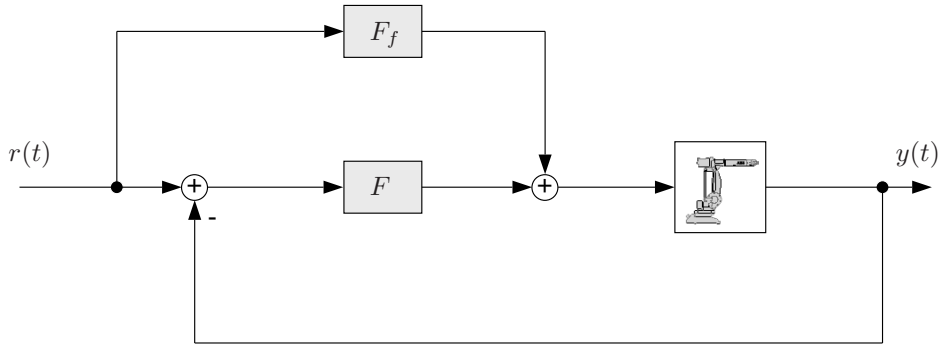


Figure 2.7 *An example of a robot controlled by feedforward and feedback controllers.*

ing the actual controller, and the third block is the robot. The feedforward part of the controller, F_f, is used for most of the control of the system. It is feeding forward a computed torque to the motors of the robot as well as doing decoupling and linearization of the system. There is a high demand on accurate models of the system to get a good feedforward part in the controller. Since there are always disturbances and modeling errors the control system has a feedback part, F. The input to the system is the motor reference which is calculated from the arm reference.

3

Physical modeling

The most important factors in manufacturing are quality, cost and productivity. When robots are used to meet these demands it is important to have good models of the robots in the controllers for fast and accurate control. To reduce the cost, the robots are built to be lighter, and this results in weaker structures. It is therefore a need for flexible robot models instead of rigid models to keep the same quality and performance of the robot.

Here direct identification of physical parameters of flexible robots are particularly interesting. A flexible robot arm can be described by a partial differential equation. However it is found that a linear model gives good results, see, for example, Spong and Vidyasagar (1989). The focus is therefore on linear approximations of the flexible robot arm.

This chapter begins with a brief introduction to modeling using Lagrangians. Then the models of a two-mass approximation and a three-mass approximation of a single link manipulator are derived, and state space equations are presented. The use of modeling tools with focus on Mathematica and Modelica will also be addressed.

3.1 Modeling with Lagrangians

There are two common ways of deriving a dynamic model of a robot. One is with the use of the so called Euler-Lagrange equations, and the other is the Newton-Euler formulation. The Euler-Lagrange equations will be discussed in this section. The material in this section is partly collected from Spong and Vidyasagar (1989); Norrlöf (1999).

The Euler-Lagrange equations are based on differential equations subject to holonomic constraints, i.e. constraints that equal zero. This can be, for example, two masses connected by a bar. If the position of the two particles are p_1 and p_2 respectively, the constraint becomes

$$|p_1 - p_2| - l = 0 \tag{3.1}$$

where l is the length of the bar. More formally let $r_1 \dots r_k$ be coordinates. Then a constraint is called holonomic if it is in the form

$$f_i(r_1, \dots, r_k) = 0 \qquad i = 1, \dots, j \tag{3.2}$$

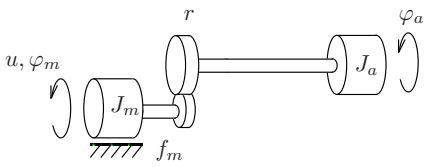
A system subject to j holonomic constraints have in general j fewer degrees of freedom than the unconstrained system.

It is often possible to give the coordinates of the system in another set of coordinates called *generalized coordinates* (φ_j), which are independent. These generalized coordinates are then used in the Euler-Lagrange equation

$$\frac{\partial}{\partial t}\frac{\partial L}{\partial \dot{\varphi}_j} - \frac{\partial L}{\partial \varphi_j} = \tau_j \tag{3.3}$$

where $L = K - V$ is called the Lagrangian, where K is the kinetic energy, and V is the potential energy.

Example 3.1 Modeling a stiff robot arm with gear box



The robot arm is affected by a torque u. The arm and gear box are stiff (no flexibilities). The movement is perpendicular to the gravitation, and therefore

the potential energy is always zero. The Lagrangian L becomes

$$K = \frac{1}{2}J_m\dot{\varphi}_m^2 + \frac{1}{2}J_a\dot{\varphi}_a^2 \tag{3.4}$$

$$V = 0 \tag{3.5}$$

$$L = K - V \tag{3.6}$$

The Euler-Lagrange equation (3.3) gives (using $r \cdot \varphi_m = \varphi_a$)

$$(J_m + r^2 J_a)\ddot{\varphi}_m - 0 = \tau \tag{3.7}$$

where $\tau = u - f_m\dot{\varphi}_m$ comes from the input torque, u, and the torque from the friction $f_m\dot{\varphi}_m$. The differential equation for the system can thus be written

$$(J_m + r^2 J_a)\ddot{\varphi}_m + f_m\dot{\varphi}_m = u \tag{3.8}$$

In general the resulting model can often be written as

$$D(\varphi)\ddot{\varphi} + C(\varphi, \dot{\varphi})\dot{\varphi} + g(\varphi) = \tau \tag{3.9}$$

where $D(\varphi)$ is the inertia matrix, τ is the motor torque, $g(\varphi)$ represents the gravity, and $C(\varphi, \dot{\varphi})\dot{\varphi}$ represents the coriolis and centrifugal forces. More information about how to calculate the inertia matrix can, for example, be found in Spong and Vidyasagar (1989). The equations for a model of the IRB 1400 used in this thesis can be found in Norrlöf (1999). The models presented in this chapter are simple, and the Euler-Lagrange notation is not necessary for the derivation.

3.2 Modeling the movement around axis one

The work is restricted to movement around axis one, see Figure 2.4. The model explains how the electrical motor torque affects the angle of the arm. The derivation of the models are based on the following assumptions: The fast controllers of the electrical motors are neglected, and the torque reference to the motors is viewed as the actual torque applied to the system. It is also assumed that only viscous friction exists. The backlash in the gear box is assumed to be neglectable, and the spring is assumed to have linear stiffness.

The robot arm can be modeled with different levels of approximation. A first approximation can be to assume that the robot arm is a rigid structure. A rigid model is investigated in Section 3.2.1. In robots having gear boxes this is a rough approximation. Modeling the gear box as a spring coupling two masses gives a more accurate model, see Section 3.2.2. If the robot is moving fast the robot arm

cannot be assumed to be stiff. This makes it reasonable to assume a three-mass model. In Section 3.2.3 the model has one more mass and spring in addition to the two-mass flexible model. This makes it a three-mass flexible model. The notations regarding the physical modeling of the robot arm can be seen in Table 3.1.

Symbol	Explanation
φ_m	motor angle
φ_g, φ_a, φ_p	arm angles
f_m	friction coefficient of the motor
k_g, k_a, k_p	spring constants
d_g, d_a, d_p	damping coefficients in the springs
J_m	moment of inertia of the motor
J_g, J_a, J_p	moments of inertia of the arm
τ	motor torque
r	gear box ratio ($\frac{1}{118}$ for axis one of IRB 1400)

Table 3.1 *Notations.*

3.2.1 Rigid model

This model, which will be denoted *rigid model,* has two masses, one represents the motor and the other the arm. They have a stiff gear box connection which makes it possible to see them as only one mass. Even though this model can be expressed with one moment of inertia we use two (J_m and J_a). This makes it easier to compare with models containing flexibilities. Often J_m is given by prior knowledge. Then it is necessary to see how the two masses are connected via the gear box. The viscous friction at the motor is denoted f_m. This model can be seen in Figure 3.1. The equation of the *rigid model* is (see Example 3.1)

$$(J_m + r^2 J_a)\ddot{\varphi}_m + f_m\dot{\varphi}_m = \tau \tag{3.10}$$

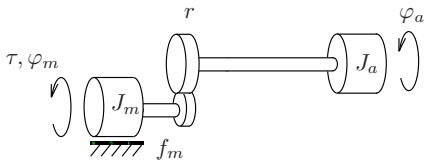


Figure 3.1 *Rigid model of the robot arm.*

The inertia of the arm is multiplied, due to the gears, with the square of the gear ratio. The ratio of the gear box for the robot used in this thesis is $r \approx 1/118$. Sometimes this modified arm inertia is used in the robot community instead of J_a, that is $\tilde{J}_a = r^2 J_a$.

3.2.2 Two-mass flexible model

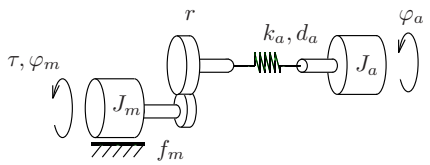


Figure 3.2 *Two-mass flexible model of the robot arm.*

The model is illustrated in Figure 3.2, using the same notation as in the rigid model with the addition of the spring constant k_a, and the damping coefficient of the spring d_a. The equations for the *two-mass flexible model* are

$$J_m\ddot{\varphi}_m + f_m\dot{\varphi}_m + r \cdot d_a(r\dot{\varphi}_m - \dot{\varphi}_a) + r \cdot k_a(r\varphi_m - \varphi_a) = \tau \tag{3.11}$$

$$J_a\ddot{\varphi}_a - d_a(r\dot{\varphi}_m - \dot{\varphi}_a) - k_a(r\varphi_m - \varphi_a) = 0 \tag{3.12}$$

These equations can be transformed into state space form. As will be pointed out in Section 5.2 it will turn out practical to use $y = \dot{\varphi}_m$ as output signal. This also means that three state variables can be used instead of four, which would be the case if φ_m was used as output. The four state model is used for control in Chapter 8. With $\dot{\varphi}_m$ as output and τ as input (u) the model can be written in state space form as

$$x(t) = \begin{pmatrix} r\varphi_m(t) - \varphi_a(t) \\ \dot{\varphi}_m(t) \\ \dot{\varphi}_a(t) \end{pmatrix} \tag{3.13a}$$

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{3.13b}$$

$$y(t) = Cx(t) \tag{3.13c}$$

$$A = \begin{pmatrix} 0 & r & -1 \\ -\frac{r \cdot k_a}{J_m} & -\frac{f_m + r^2 d_a}{J_m} & \frac{r \cdot d_a}{J_m} \\ \frac{k_a}{J_a} & \frac{r \cdot d_a}{J_a} & -\frac{d_a}{J_a} \end{pmatrix} \tag{3.13d}$$

$$B = \begin{pmatrix} 0 \\ \frac{1}{J_m} \\ 0 \end{pmatrix} \qquad C = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} \tag{3.13e}$$

Using Laplace transformation and $\mathcal{L}\{\varphi_m(t)\} = \Phi_m(s)$ the relation between torque and motor velocity can be written as

$$s\Phi_m(s) = G_{\tau m_v}(s)\tau(s)$$

$$G_{\tau m_v}(s) = \frac{J_a s^2 + d_a s + k_a}{s^3 J_a J_m + s^2(J_a f_m + d_a(J_m + r^2 Ja)) + s(k_a(J_m + r^2 J_a) + d_a f_m) + k_a f_m} \tag{3.14}$$

If it is assumed that the effect of the damping coefficient, d_a, is negligible the transfer function becomes

$$G_{\tau m_v}(s) \approx \frac{J_a s^2 + k_a}{s^3 J_a J_m + s^2 J_a f_m + s \cdot k_a (J_m + r^2 J_a) + k_a f_m} \tag{3.15}$$

We can also get a transfer function from Laplace transforming (3.12). With $\mathcal{L}\{\varphi_a(t)\} = \Phi_a(s)$, the relationship from motor angle to arm angle is

$$\Phi_a(s) = \frac{(sd_a + k_a)r}{J_a s^2 + d_a s + k_a} \Phi_m(s) = G_a(s)\Phi_m(s) \tag{3.16}$$

Note that the poles of this transfer function appears as zeros of the transfer function between the torque and motor velocity in (3.14). The Laplace transformation of (3.11) gives

$$\begin{aligned} \Phi_m(s) &= \frac{1}{s^2 J_m + s(r^2 d_a + f_m) + k_a r^2} (r(sd_a + k_a)\Phi_a(s) + \tau(s)) \\ &= G_m(s)(G_1(s)\Phi_a(s) + \tau(s)) \end{aligned} \tag{3.17}$$

With G_a from (3.16) and G_m and G_1 from (3.17) we can build the block diagram in Figure 3.3.

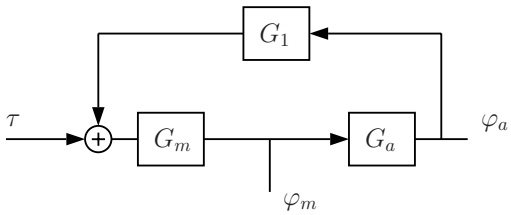


Figure 3.3 *Block diagram of the two-mass flexible model.*

3.2.3 Three-mass flexible model

The model is illustrated in Figure 3.4. Denote the mass after the gear box (the middle mass) with subscript g. The equations are

$$\begin{aligned} J_m\ddot{\varphi}_m + f_m\dot{\varphi}_m + r \cdot d_g(r\dot{\varphi}_m - \dot{\varphi}_g) + r \cdot k_g(r\varphi_m - \varphi_g) &= \tau \\ J_g\ddot{\varphi}_g - d_g(r\dot{\varphi}_m - \dot{\varphi}_g) - k_g(r\varphi_m - \varphi_g) + d_a(\dot{\varphi}_g - \dot{\varphi}_a) + k_a(\varphi_g - \varphi_a) &= 0 \\ J_a\ddot{\varphi}_a - d_a(\dot{\varphi}_g - \dot{\varphi}_a) - k_a(\varphi_g - \varphi_a) &= 0 \end{aligned} \tag{3.18}$$

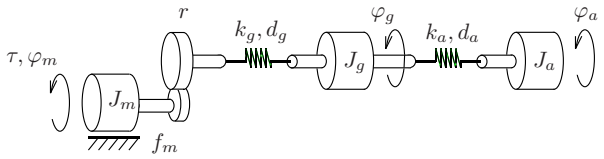


Figure 3.4 *Three-mass flexible model of the robot arm.*

In state space notation (3.18) gives

$$x(t) = \begin{pmatrix} r\varphi_m(t) - \varphi_g(t) \\ \varphi_g(t) - \varphi_a(t) \\ \dot{\varphi}_m(t) \\ \dot{\varphi}_g(t) \\ \dot{\varphi}_a(t) \end{pmatrix} \tag{3.19a}$$

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{3.19b}$$

$$y(t) = Cx(t) \tag{3.19c}$$

$$A = \begin{pmatrix} 0 & 0 & r & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ -\frac{r \cdot k_g}{J_m} & 0 & -\frac{r^2 d_g + f_m}{J_m} & \frac{r \cdot d_g}{J_m} & 0 \\ \frac{k_g}{J_g} & -\frac{k_g}{J_g} & \frac{r d_g}{J_g} & -\frac{d_g + d_a}{J_g} & \frac{d_a}{J_g} \\ 0 & \frac{k_a}{J_a} & 0 & \frac{d_a}{J_a} & -\frac{d_a}{J_a} \end{pmatrix} \tag{3.19d}$$

$$B = \begin{pmatrix} 0 & 0 & \frac{1}{J_m} & 0 & 0 \end{pmatrix}^T \tag{3.19e}$$

$$C = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix} \tag{3.19f}$$

3.3 Mathematical and modeling tools

The more complex the model is the bigger the need is for mathematical and modeling tools. In Chapter 5 MATLAB™ and the System Identification Toolbox (SITB) are used for identifying parameters of the models. In the modeling work the language Modelica can be used.

Modelica is a standardized modeling language (Otter and Elmqvist, 2001; Tiller, 2001), but it does not include simulation or analysis capabilities. For that purpose MathModelica and Dymola have been built, see Jirstrand (2000) and Dymola

(2002) respectively. MathModelica is an add-on to Mathematica (Wolfram, 1999), and can therefore use the vast capabilities of Mathematica in analysis and manipulation. MathModelica has a graphical editor, which is an extension to Microsoft software Visio for diagram design. In the graphical editor it is possible to choose components from the standard libraries, and then connect them. One can also enter parameter values for different components. MathModelica has also simulation capabilities. It uses the Dymola engine to simulate the models. The graphical representation of the models can be converted to textual models in a Mathematica notebook. The notebook can be used for documenting and storing but also for model analysis, transfer function computations, etc. More information about modeling robot arms using MathModelica can be found in Östring et al. (2002b); Norrlöf et al. (2002).

3.3.1 Physical modeling using MathModelica

As said in Section 3.2, the robot arm can be approximated with several coupled masses. The more masses the more complex the model becomes. Even a three-mass model is fairly large, see (3.19). It is therefore useful to have a tool helping with the modeling.

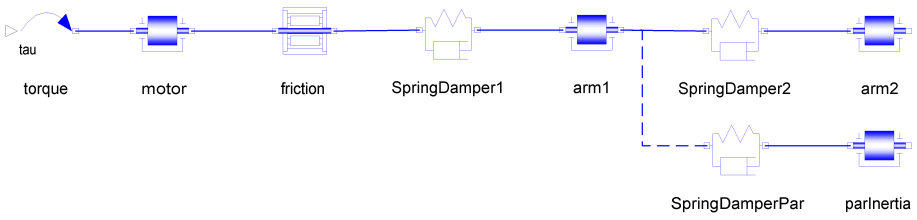


Figure 3.5 *The Modelica model of three- and four-mass models. The extension from the three-mass model to the four-mass model is shown as a dashed connection.*

In Figure 3.5 the MathModelica implementation of a three-mass and a four-mass model is shown. This section describes the generation of the equations for the three-mass model from the graphical representation. The four-mass model is generated in a similar way.

From the graphical description of the system in Figure 3.5 it is possible to make a simulation using the MathModelica environment. This gives a simulated model

with 6 continuous states. By directly taking the states from the Modelica model, using the MathModelica command `GetFlatStateVariables[ThreeMM]`, a total of 9 states are found. The difference in the number of states comes from the fact that the different sub-models are modeled individually, and when they are interconnected it is possible to reduce the total number of states. In Mathematica, using MathModelica, the equations describing the three-mass system can be found with the command, `GetFlatEquations[ThreeMM]`. It results in 55 equations. Many are trivial, e.g. saying that the angular velocity on one edge of a component equals the angular velocity of the connecting edge of the next component.

The trivial equations can be removed in Mathematica using `Eliminate[eqs,list]` where `eqs` are the equations coming from the Modelica description, and `list` is a list of the variables that should be eliminated. The result from this step is that the number of equations are reduced from 55 to 6. In the current version of MathModelica the list of variables to eliminate must be found by hand.

After this step it is also necessary to find and replace some state variables that can be expressed as functions of other state variables. For the three-mass model this includes the spring damper components where the relative angular position between the two connections becomes a state variable. These two variables are replaced by $\varphi_m r - \varphi_g$ and $\varphi_g - \varphi_a$, respectively.

The next step is to introduce the state variables, x_1 to x_5. For the three-mass model described here they are given by

$$\begin{aligned} x_1 &= \varphi_m r - \varphi_g, \quad x_2 = \varphi_g - \varphi_a, \\ x_3 &= \dot{\varphi}_m, \quad x_4 = \dot{\varphi}_g, \quad x_5 = \dot{\varphi}_a \end{aligned} \tag{3.20}$$

For Mathematica to be able to solve the equations it is also necessary to include the time derivative of the state variables among the equations added to the 6 equations found above. This step is done with the command `Join[list1,list2]` in Mathematica. `Solve[eqs,{x1',x2',x3',x4',x5'}]` finally gives the state equations for the three-mass model in a closed form.

The state space description of the three-mass model from Mathematica becomes

$$
\begin{aligned}
A &= \begin{pmatrix}
0 & 0 & r & -1 & 0 \\
0 & 0 & 0 & 1 & -1 \\
-\frac{k_g r}{J_m} & 0 & -\frac{f_m + d_g r^2}{J_m} & \frac{d_g r}{J_m} & 0 \\
\frac{k_g}{J_g} & -\frac{k_a}{J_g} & \frac{d_g r}{J_g} & -\frac{d_g + d_a}{J_g} & \frac{d_a}{J_g} \\
0 & \frac{k_a}{J_a} & 0 & \frac{d_a}{J_a} & -\frac{d_a}{J_a}
\end{pmatrix} \\
B &= \begin{pmatrix} 0 & 0 & \frac{1}{J_m} & 0 & 0 \end{pmatrix}^T \\
C &= \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}
\end{aligned}
\tag{3.21}
$$

Output from this model is the angular velocity of the motor. If the angular position is chosen as output the number of states becomes 6 since an extra integrator has to be added. This is equal to the simulation model in MathModelica.

An advantage using MathModelica is that it is very easy to extend a model. Here a four-mass model with the extra parallel mass according to Figure 3.5 is shown. In addition to parameters and variables in of the three-mass model, the additional parameters are J_p, k_p, and d_p, and the additional variable is φ_p. The state space model from Mathematica becomes

$$
\begin{aligned}
A &= \begin{pmatrix}
0 & 0 & 0 & r & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & -1 \\
-\frac{k_g r}{J_m} & 0 & 0 & -\frac{f_m + d_g r^2}{J_m} & \frac{d_g r}{J_m} & 0 & 0 \\
\frac{k_g}{J_g} & -\frac{k_a}{J_g} & -\frac{k_p}{J_g} & \frac{d_g r}{J_g} & -\frac{d_g + d_a + d_p}{J_g} & \frac{d_a}{J_g} & \frac{d_p}{J_g} \\
0 & \frac{k_a}{J_a} & 0 & 0 & \frac{d_a}{J_a} & -\frac{d_a}{J_a} & 0 \\
0 & 0 & \frac{k_p}{J_p} & 0 & \frac{d_p}{J_p} & 0 & -\frac{d_p}{J_p}
\end{pmatrix} \\
B &= \begin{pmatrix} 0 & 0 & 0 & \frac{1}{J_m} & 0 & 0 & 0 \end{pmatrix}^T \\
C &= \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}
\tag{3.22}
$$

where the states are chosen as

$$
\begin{aligned}
&x_1 = \varphi_m r - \varphi_g, \quad x_2 = \varphi_g - \varphi_a, \\
&x_3 = \varphi_g - \varphi_p, \quad x_4 = \dot{\varphi}_m, \\
&x_5 = \dot{\varphi}_g, \quad x_6 = \dot{\varphi}_a, \quad x_7 = \dot{\varphi}_p
\end{aligned}
\tag{3.23}
$$

Here $\dot{\varphi}_m$ is chosen as output. With the angular position as output, φ_m, the number of states becomes 8.

The models in (3.21) and (3.22) are symbolically represented in Mathematica, and this makes it very easy to reparameterize and scale the parameters. After this step

the model description is saved in a text file. This file can then be transformed, using a simple MATLAB™ program, into an m-file (MATLAB™ executable file) that can be used directly by the System Identification Toolbox (SITB). Using these tools the modeling to identification process becomes straight-forward and almost automatic.

4

Identification methods

Estimating models from measured data is what identification is all about. This chapter deals with basic principles regarding prediction error identification methods in general and using data collected in closed loop in particular.

First some model structures are presented in Section 4.1. In Section 4.2 different ways of computing the estimate are discussed. Non-parametric methods are discussed in Section 4.3, and aspects related to closed loop data are treated in Section 4.4. The material presented in this chapter is standard, and could be skipped by readers who are familiar with the topics. The fundamentals of this subject are presented in, for example, Ljung and Glad (1994). A more detailed treatment of the topics can be found in Ljung (1999) and in Söderström and Stoica (1989).

4.1 Model structures

The kind of systems which are studied here are usually linear time-invariant. A vector of parameters is adjusted so that the model mimics the real system. A set of models specified by certain parameters is called a *model class*. It includes both

a model of the input-output dynamics as well as the noise.

Consider the following model structure given in discrete time and parameterized by θ

$$y(t) = G(q,\theta)u(t) + H(q,\theta)e(t) \tag{4.1}$$

The model $G(q,\theta)$ will be called the system model and $H(q,\theta)$ the noise model. The shift operator is denoted by q, $qu(t) = u(t+1)$. The noise model is driven by white noise $e(t)$. To identify the parameters, θ, in this model structure, usually some criterion function is minimized

$$\hat{\theta} = \arg\min_{\theta} V_N(\theta, Z_N) \tag{4.2}$$

where Z_N denotes the data set. The criterion function is often called loss function. The criterion function (4.2) is usually formed using the one-step-ahead predictor of $y(t)$ given measurements up to time $t-1$. Equation (4.1) can be rewritten as

$$H^{-1}(q,\theta)y(t) = H^{-1}(q,\theta)G(q,\theta)u(t) + e(t) \tag{4.3}$$

and

$$y(t) = H^{-1}(q,\theta)G(q,\theta)u(t) + (1 - H^{-1}(q,\theta))y(t) + e(t) \tag{4.4}$$

White noise cannot be predicted, and by removing $e(t)$ the predictor can be written as

$$\hat{y}(t,\theta) = H^{-1}(q,\theta)G(q,\theta)u(t) + (1 - H^{-1}(q,\theta))y(t) \tag{4.5}$$

Note that the predictor only depends on old outputs $y(t-1),\ y(t-2), \cdots$ since H is assumed to be monic. A common criterion function is

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon^2(t,\theta) = \frac{1}{N}\sum_{t=1}^{N} (y(t) - \hat{y}(t,\theta))^2 \tag{4.6}$$

In general it is possible to use any arbitrary positive, scalar-valued function $l(\varepsilon)$ as measure

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N} l(\varepsilon(t,\theta)) \tag{4.7}$$

Other norms than $l(\varepsilon) = \varepsilon^2$ might be useful, for example, for robustness against outliers (Ljung, 1999).

In most cases it is convenient and sufficient to use some standard model structure. In black-box modeling with prediction error methods the following general model structure is often used

$$A(q)y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t) \tag{4.8}$$

where

$$A(q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a} \tag{4.9}$$

and similarly for the C-, D-, and F-polynomials, while

$$B(q) = b_1 q^{-1} + b_2 q^{-2} + \cdots + b_{n_b} q^{-n_b} \tag{4.10}$$

Some common model structures that are special cases of (4.8) are shown in Table 4.1.

Polynomials used	Name of the model structure
B	FIR (finite impulse response)
A, B	ARX
A, B, C	ARMAX
B, F	OE (output error)
B, C, D, F	BJ (Box-Jenkins)

Table 4.1 *Some common model structures.*

Another way of parameterizing the models is to use physical parameters. This is natural when physical modeling is used. It is often called grey-box models. Physical modeling will benefit from using a model structure that is adjusted to the system. In some situations the value of the physical parameters is the objective, for example, when monitoring a physical parameter. Compared to the black-box models the physical models are almost always in continuous time. The parameters in the physical models can appear quite complicated in the model. Sometimes it is possible to make a variable change to get a less complex model to identify. These kinds of models are frequently used in this thesis and are described in more detail in Chapters 3, 5, and 6.

4.2 Computing the estimate

If an FIR or an ARX model is used together with a quadratic criterion function the estimate is found using standard least squares method. As seen in Table 4.1 the ARX model structure has the following parameterization

$$G(q, \theta) = \frac{B(q, \theta)}{A(q, \theta)} \tag{4.11}$$

$$H(q, \theta) = \frac{1}{A(q, \theta)} \tag{4.12}$$

The simplicity of this model structure is revealed by that the one-step-ahead predictor can be written as

$$\hat{y}(t,\theta) = \varphi(t)\theta \tag{4.13}$$

$$\varphi(t) = [-y(t-1)\cdots - y(t-n_a)\ u(t-n_k)\cdots u(t-n_k-n_b+1)]^T \tag{4.14}$$

$$\theta = [a_1 \cdots a_{n_a} b_1 \cdots b_{n_b}] \tag{4.15}$$

where n_k is the number of delays in the system. This means that this model structure gives a predictor which is linear in parameters, and the problem of finding the solution to Equation (4.6) is solved using standard least squares

$$\hat{\theta} = \arg\min_{\theta} V_N(\theta) = \left[\frac{1}{N}\sum_{t=1}^{N}\varphi(t)\varphi(t)^T\right]^{-1}\frac{1}{N}\sum_{t=1}^{N}\varphi(t)y(t) \tag{4.16}$$

which is usually computed using QR-factorization (see again Ljung, 1999)

For other parameterizations and criteria a numerical search scheme is applied to find the estimate. A standard choice is to use a search routine of the form

$$\hat{\theta}_{(i+1)} = \hat{\theta}_{(i)} - \mu_{(i)}[R_{(i)}]^{-1}V_N'(\hat{\theta}_{(i)}, Z_N) \tag{4.17}$$

where $V_N'(\hat{\theta}_{(i)}, Z_N)$ denotes the gradient of the criterion function $V_N(\hat{\theta}_{(i)}, Z_N)$ with respect to θ, $R_{(i)}$ is a matrix that modifies the search direction, and $\mu_{(i)}$ is a scaling factor which determines the step length.

The criterion function

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{2}\varepsilon^2(t,\theta) = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{2}(y(t)-\hat{y}(t,\theta))^2 \tag{4.18}$$

gives

$$V_N'(\theta) = \frac{1}{N}\sum_{t=1}^{N}\psi(t,\theta)\varepsilon(t,\theta) \tag{4.19}$$

where $\psi(t,\theta)$ denotes the negative gradient of the prediction error

$$\psi^T(t,\theta) = -\frac{d}{d\theta}\varepsilon(t,\theta) = \frac{d}{d\theta}\hat{y}(t,\theta) \tag{4.20}$$

Common choices of $R_{(i)}$ are the Hessian $V''(\theta)$ or approximations of $V''(\theta)$. One approximation of the Hessian is

$$R_{(i)} = \frac{1}{N}\sum_{t=1}^{N}\psi(t,\theta)\psi^T(t,\theta) \tag{4.21}$$

which gives the Gauss-Newton method. To get rid of problems related to $R_{(i)}$ being close to singular a term λI can be added. This way of doing regularization is called the Levenberg-Marquardt procedure.

4.3 Non-parametric frequency domain identification

Non-parametric means that the transfer function is estimated without using a certain model set described by a number of parameters. Instead the value of the transfer function is estimated at each frequency. The estimate is very crude, and in order to be useful it has to be smoothed to reduce the variance. A parameter that determines the smoothing is chosen. The smoothing operation is achieved by weighting nearby frequencies.

4.3.1 The empirical transfer function estimate (ETFE)

To begin with the Fourier transform of the input and the output is

$$U_N(\omega) = \frac{1}{\sqrt{N}} \sum_{t=1}^{N} u(t) e^{-i\omega t} \tag{4.22}$$

$$Y_N(\omega) = \frac{1}{\sqrt{N}} \sum_{t=1}^{N} y(t) e^{-i\omega t} \tag{4.23}$$

The empirical transfer function estimate (ETFE) is defined as

$$\hat{\hat{G}}_N(e^{i\omega}) = \frac{Y_N(\omega)}{U_N(\omega)} \tag{4.24}$$

The frequency response is estimated as the ratio of the output and the input Fourier transforms. It is possible to show that the variance of the estimate does not approach zero as the number of data tends to infinity (see Ljung, 1999). A weighting function is then used to smooth the estimate

$$\hat{G}_N(e^{i\omega}) = \frac{\sum_k \alpha_k(\omega) \hat{\hat{G}}_N(e^{i\omega_k})}{\sum_k \alpha_k(\omega)} \tag{4.25}$$

This smoothing operation reduces the variance but introduces a bias. The width of this smoothing function is a parameter that is specified by the user.

4.3.2 Spectral estimation (SPA)

Using the Fourier transform above the periodogram $\hat{\hat{\Phi}}_N(\omega)$ is defined as

$$\hat{\hat{\Phi}}_u(\omega) = |U_N(\omega)|^2 \tag{4.26}$$

and similarly for $\hat{\hat{\Phi}}_{yu}(\omega)$

$$\hat{\hat{\Phi}}_{yu}(\omega) = Y_N(\omega)\bar{U}_N(\omega) \tag{4.27}$$

To do the spectral estimation of the transfer function first the periodogram is smoothed to get an estimate of the spectrum. Then the estimate of the transfer function (SPA) is found using the smoothed estimates $\hat{\Phi}_u$ and $\hat{\Phi}_{yu}$ as

$$\hat{G}(e^{i\omega}) = \frac{\hat{\Phi}_{yu}(\omega)}{\hat{\Phi}_u(\omega)} \tag{4.28}$$

4.3.3 Non-parametric estimation during closed loop

Let the data come from a closed loop system

$$\begin{aligned} y(t) &= G_0(q)u(t) + v(t) \\ u(t) &= r(t) - F_y(q)y(t) \end{aligned} \tag{4.29}$$

where $F_y(q)$ denotes the controller, and $G_0(q)$ denotes the true system. When doing empirical transfer function estimate or a spectral estimation in closed loop there will be a bias in the estimates. The estimates of the spectra are affected by the feedback. Let the spectral estimate be denoted $\hat{G}(e^{i\omega})$, then the estimate tends to (Ljung, 1999)

$$\hat{G}(e^{i\omega}) = \frac{G_0(e^{i\omega})\Phi_r(\omega) - F_y(e^{-i\omega})\Phi_v(\omega)}{\Phi_r(\omega) + |F_y(e^{i\omega})|^2\Phi_v(\omega)} \tag{4.30}$$

as the number of data goes to infinity and where Φ_r and Φ_v denotes the spectrum of $r(t)$ and $v(t)$. If there is no noise in the system the estimate will tend to the true system. On the other hand if the noise dominates over the reference signal the estimate of the system will tend to the inverse of the controller.

4.4 Data collected in closed loop

In the problem studied in this thesis the data collection from the flexible robot is performed during feedback. The main problem with closed loop identification is that the data contain less information about the open loop system. This is because the purpose of the feedback is to make the closed loop system less sensitive to changes in the open loop system.

There are three main approaches to closed loop identification (Ljung, 1999); Indirect approach, Joint input-output approach, and Direct approach.

Indirect approach. Initially the closed loop transfer function is identified. The open loop transfer function is then calculated using the known controller, which should be linear in order to be able to use a linear model when estimating the closed loop transfer function. This is not possible in the application here because the controller is not known.

Joint input-output approach. In this approach the controller is estimated. This is difficult if the controller is nonlinear or time variant.

Direct approach. The open loop system is identified using measurements of the input and output ignoring the feedback. It is straightforward to apply a basic prediction error method if we bare in mind some properties from closed loop identification. We have chosen the direct approach here. Prediction error methods are also well investigated, and there are several software packages available.

In the following sections different aspects of prediction error methods applied to closed loop data are studied. Section 4.4.1 shows the requirements for consistency. The bias contribution is studied in Section 4.4.2, and the variance of the estimates is discussed in Section 4.4.3.

For the interested reader the subject of closed loop identification is more thoroughly treated in, for example, Forssell (1999).

4.4.1 The system is included in the model class

First some notations and background will be introduced. To start with the difference between two models is studied. Is it possible to separate two different models when applied to a certain input signal? Instead of looking at the system and noise model it is possible to only consider the predictor filters, $W_y(q)$ and $W_u(q)$ in

$$\begin{aligned}\hat{y}(t,\theta) &= H^{-1}(q,\theta)G(q,\theta)u(t) + (1 - H^{-1}(q,\theta))y(t) \\ &= W_u(q,\theta)u(t) + W_y(q,\theta)y(t)\end{aligned} \tag{4.31}$$

Let the two models be denoted by subscript 1 and 2, and study the difference between them

$$W_{u,1} - W_{u,2} = \Delta W_u \tag{4.32}$$

$$W_{y,1} - W_{y,2} = \Delta W_y \tag{4.33}$$

A data set is *informative enough* (see Ljung, 1999) with respect to the model set if

$$\bar{E}[\Delta W_u u(t) + \Delta W_y y(t)]^2 = 0 \tag{4.34}$$

implies that $W_{u,1}(e^{i\omega}) \equiv W_{u,2}(e^{i\omega})$ and $W_{y,1}(e^{i\omega}) \equiv W_{y,2}(e^{i\omega})$ for almost all frequencies. The symbol $\bar{E}$ is defined as

$$\bar{E}f(t) = \lim_{N\to\infty} \frac{1}{N} \sum_{t=1}^{N} Ef(t) \tag{4.35}$$

A signal is called *persistently exciting* if the spectrum of the signal is different from zero for almost all frequencies.

In Ljung (1999) the following main result is presented: Prediction error methods will consistently estimate the system, regardless if the data have been collected under feedback or not if:

1. The data are informative.
2. The model set contains *the true* system.

Closed loop experiments are informative with respect to the model class of all linear systems if and only if the reference signal is persistently exciting, that is the spectrum should include energy for almost all frequencies. Except from having a good data set we need a good model set. It is, in all real applications, impossible to have a model set that contains the true system, but it is important that we are as close as possible. Note that open loop data give unbiased estimates if OE-models are used. This is *not* true for closed loop data.

We are now ready to take a closer look at the system subject to feedback. Let the true system be written as

$$y(t) = G_0(q)u(t) + H_0(q)e(t) \tag{4.36}$$

where G_0 and H_0 denote the true system. Let it be subject to feedback using a linear controller $F_y(q)$. The input and output signals are driven by the reference signal $r(t)$ and the white noise $e(t)$

$$y(t) = G_0(q)S_0(q)r(t) + S_0(q)H_0(q)e(t) \tag{4.37}$$

$$u(t) = S_0(q)r(t) - F_y(q)S_0(q)H_0(q)e(t) \tag{4.38}$$

where

$$S_0(q) = \frac{1}{1 + F_y(q)G_0(q)} \tag{4.39}$$

The condition (4.34) can be written

$$0 = \bar{E}\left|[\Delta W_y \quad \Delta W_u]\begin{bmatrix} y \\ u \end{bmatrix}\right|^2 = \bar{E}\left|[\Delta W_y \quad \Delta W_u]\begin{bmatrix} G_0 & S_0 \\ 1 & -F_yS_0 \end{bmatrix}\begin{bmatrix} S_0r \\ H_0e \end{bmatrix}\right|^2 \tag{4.40}$$

The determinant of the second matrix is $-G_0F_yS_0 - S_0 = -1$, thus it is invertible. In the last matrix $H_0e(t)$ is persistently exciting. If $S_0r(t)$ is that too, the conclusion is that $\Delta W_y = 0$ and $\Delta W_u = 0$. $S_0r(t)$ is persistently exciting if $r(t)$ is. S_0 can only cancel a finite number of frequencies.

Let the model be of order m and included in the model class. Let $r(t)$ only be persistently exciting of some order $k \gg m$. It can typically consist of a large number of sinusoids. Then $S_0(q)$ can only cancel a finite number of sinusoids. In (4.37) it can be seen that G_0 can be uniquely identified if the number of sinusoids is large enough, because it is possible to uniquely identify a system if the input consists of a number of sinusoids, if the number is greater than the order of the system.

Hence the conclusion is that it is possible to use a reference signal consisting of a sum of sinusoids for closed loop identification if the number of sinusoids is high enough.

4.4.2 The system is not included in the model class

When the system is not included in the model class the model will be biased. The bias will be discussed below, using the theory from Ljung (1999).

All the calculations in this section are carried out under the assumption that the number of data tends to infinity. Let G_0 and H_0 denote the true system, and let the model structure be given by

$$y(t) = G(q,\theta)u(t) + H(q,\theta)e(t)$$

Because of the closed loop the input spectrum Φ_u can be split into two parts originating from the reference r and the noise e (variance λ_0):

$$\Phi_u(\omega) = \Phi_u^r(\omega) + \Phi_u^e(\omega)$$

The bias in the parameter estimates can be seen in

$$[\hat{G},\, \hat{H}] = \arg\min_{\theta} \int_{-\pi}^{\pi} |G_0(e^{i\omega}) + B(e^{i\omega},\theta) - G(e^{i\omega},\theta)|^2 \frac{\Phi_u(\omega)}{|H(e^{i\omega},\theta)|^2} d\omega + \lambda_0 \int_{-\pi}^{\pi} \frac{|H_0(e^{i\omega}) - H(e^{i\omega},\theta)|^2}{|H(e^{i\omega},\theta)|^2} \frac{\Phi_u^r(\omega)}{\Phi_u(\omega)} d\omega \tag{4.43}$$

where the bias term B is

$$|B(e^{i\omega},\theta)|^2 = \frac{\lambda_0}{\Phi_u(\omega)} \frac{\Phi_u^e(\omega)}{\Phi_u(\omega)} |H_0(e^{i\omega}) - H(e^{i\omega},\theta)|^2$$

The bias term B is small if

1. $|H_0(e^{i\omega}) - H(e^{i\omega}, \theta)|$ is small
2. $\Phi_u^e(\omega)/\Phi_u(\omega)$ is small
3. $\lambda_0/\Phi_u(\omega)$ is small

for all ω that are important for the model.

These conditions implies that is is preferable to have a good noise model, weak feedback and low noise level. In the application in this thesis the two last ones are not possible to affect directly. There is feedback, and there is also noise. Thus it is preferable to make a good noise model to get an unbiased estimate of G_0. We should therefore have a flexible, preferably, independently parameterized noise model. If we have an independently parameterized noise and system model the second term in (4.43) does not affect the bias of $\hat{G}$. This also suggests that the best way to estimate a low order model is to first estimate a high order model so that the true system is approximately included in the model set, and then to perform model reduction (Tjärnström, 2002).

4.4.3 Variance of the estimated model

The results in this section are asymptotic in the number of data points N and model order n. The variance of the calculated model is (see Ljung, 1999)

$$Cov\, \hat{G}(e^{i\omega}) = \frac{n}{N} \frac{\Phi_v(\omega)}{\Phi_u^r(\omega)}$$

where

$$\Phi_v(\omega) = |H_0(e^{i\omega})|^2 \Phi_e(\omega)$$

It can be seen that the part of the input that originates from the feedback signal has no influence on the variance of the estimate. The part of the feedback signal that originates from the reference signal can be written

$$\begin{aligned} \Phi_u^r(\omega) &= |S_0(e^{i\omega})|^2 \Phi_r(\omega) \\ S_0(e^{i\omega}) &= \frac{1}{1 + F_y(e^{i\omega}) G_0(e^{i\omega})} \end{aligned} \tag{4.47}$$

For those ω where $S_0(e^{i\omega})$ is small the estimate of the system gets a high variance.

5

Off-line identification

System identification is an established modeling tool in engineering, and numerous successful applications have been reported. The theory is well developed for linear time invariant systems, see, for example, Ljung (1999) or Söderström and Stoica (1989), and there are powerful software tools available, like, for example, the System Identification Toolbox for MATLAB™ (Ljung, 2000). Industrial robots represent an interesting challenge for system identification methods, and an overview of identification in robotics can be found in Kozlowski (1989).

One application area for system identification within robotics is identification of the parameters in the kinematic description of the robot, while a second area deals with the problem of identifying the parameters in the dynamical model of the robot and is often divided into rigid body and flexible body dynamics. A third area is to determine the parameters on the joint level including friction, motor characteristics, etc. Recent results from the last two areas can be found in, for example, Wang et al. (1996), Grotjahn et al. (2001) and Gautier and Poignet (2001). In these papers it is assumed that the robot is rigid.

In the work to be presented here the goal is to study identification of robots including flexibilities, and also to identify the stiffnesses and dampings for the mechan-

ical flexibilities. This topic has been addressed in, for example, Dépincé (1998), ElMaraghy et al. (1994), Nissing and Polzer (2000), Albu-Schäffer and Hirzinger (2001), Berglund and Hovland (2000), Hovland et al. (2001), Johansson et al. (2000), Pham et al. (2001) and Östring et al. (2001b). The problem considered in Berglund and Hovland (2000) and Hovland et al. (2001) is closely related to the work reported below, but the proposed solution is based of frequency domain identification in combination with the solution of an eigenvalue problem. In Johansson et al. (2000) time domain identification methods of black-box type are applied, which means that no physical parameters are obtained directly from the identification. The work presented in Albu-Schäffer and Hirzinger (2001) deals with identification of a lightweight robot with seven degrees of freedom, and the results involve identification of joint elasticity and damping parameters. These are found by applying external excitation on one axis at a time. Also in Pham et al. (2001) the identification experiments are carried out by moving one axis at a time. The sought physical parameters are obtained as nonlinear functions of the estimates obtained using a model structure that is linear in the parameters.

In the work presented in this chapter, which is an extension to the work reported in Östring et al. (2001b), the aim is to apply a method where inertial parameters as well as parameters describing the flexibility can be identified directly in the time domain. This is done by utilizing a user-defined model structure in the System Identification Toolbox.

The presented work is carried out under some simplifying conditions. First, only movements around axis one are considered. Second, all experiments are carried out with the other axes in one position. Third, only a linear model structure is considered, which means that, for example, only viscous friction is included in the model. It should be noted that although the model is linear for fixed parameters, it is not linear in the parameters. The simplifying assumptions can be motivated in different ways. First, the presented work can be seen as a feasibility study carried out in order to see whether this is a possible approach or not, and to try to reach as far as possible with linear models. In a number of the references cited above the identification experiments are carried out on one axis at a time, which indicates that this assumption is rather common. The importance of the restriction that the identification is carried out in only one operating point is related to the intended use of the model. There are at least two possible uses of the identified model. The first is to use the model for control design on joint level, and the second is to use the model for diagnosis purposes. In both cases two ways to extend the approach can be considered. One way is to identify linear time-invariant models in a number of operating points, and use gain scheduling in the control or diagnosis functions. A second alternative is to move to a nonlinear model where nonlinearity due to

variations in operating point is captured. These extensions are left for future work.

A detailed description of the robot used in the experiments can be found in Chapter 2. The robot is a commercial robot, and this means, among other things, that is not possible to alter the controller in any major way or even find out exactly how the controller works. The robot system is depicted in Figure 5.1. It is only possible to directly affect the reference signal, φ_{ref}, in this setup, not the torque, u, generated by the electrical motor.

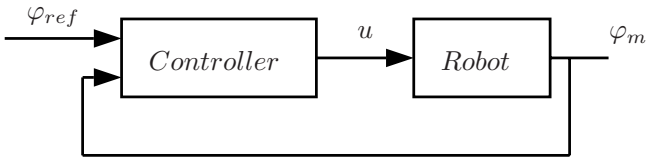


Figure 5.1 *Block diagram of the control system.*

We measure the reference signal, the torque reference from the controller, and the angle of the robot arm with a sampling interval of 0.5 msec. An electrical motor and a gear box drive the movement of the joint. The motor itself contains an inner fast controller loop for control of the motor torque. The goal is to identify a model of the robot by measuring the input signal (the torque reference) and the output signal (measured motor angle).

This chapter is organized as follows: Before the identification experiments in Section 5.2 and 5.3, the data collection is described in Section 5.1. In Section 5.4 another data set is used to look at the quality of the models, and in Section 5.5 a summary is given.

5.1 Data collection and pre-processing

The initial robot pose is with all motors in sync. This means that the robot looks like an L turned upside down, see Figure 5.2. As said previously, only the motor at the base of the robot is moving. The data are acquired with a special interface between the robot control system, S4C and MATLAB™, (see Norrlöf, 2000).

The system is under feedback. This means that it is straight forward to excite the system by applying a reference signal. It is especially important that this signal is exciting enough since the feedback may reduce the signal energy at certain frequencies, see (4.47). The reference signal, that is chosen, is a chirp signal covering 0.5-30Hz with constant amplitude, see Dépincé (1998) for another example using a

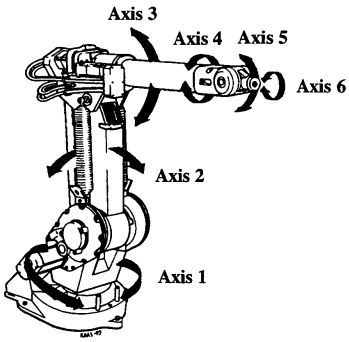


Figure 5.2 *The manipulator IRB 1400.*

chirp. The signal spans from high frequencies to low frequencies. This signal is used because it is easy to see which amplitudes at which frequencies that are excited. It is possible to make a rough amplitude Bode plot by just looking at the signals. It is also easy to see how the robot and the controller alter the reference signal to the torque reference, which is used as the input signal to the robot. An alternative way of choosing the reference signal is to use optimization of some excitation measure. In, for example, Swevers et al. (1997), a finite sum of sinusoids is proposed as input signal.

The sample rate is chosen to be the same as the sample rate of the controller. Then, before the identification, the data are re-sampled to the desired sample rate. We know the approximate frequencies of the peaks in the Bode plot, and this gives us an approximate sample rate. Then we test empirically to get as good sample rate as possible. This results in a down-sampling with a factor of ten from the original sample rate of 2000 Hz before the identification. The desired sample rate is related to the bandwidth of the system. A rule of thumb is to choose the sample rate as 10 times the bandwidth of interest for the modeling (Ljung and Glad, 1994, page 268). As a final step in the pre-processing the sample means are removed.

5.2 Black-box models

Section 4.4 indicates that a black-box model that has a flexible and independently parameterized noise model is preferred. In this section we have therefore chosen Box-Jenkins models as the model structure to work with. Box-Jenkins models can

be represented, see Chapter 4, by

$$y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t)$$

where B, C, D and F are parameterized polynomials. $u(t)$ is the input, and $e(t)$ is the noise. We can choose the orders of the polynomials and the time delay of the input signal. These are often denoted as five numbers in a vector, $\mathcal{N} = [n_b\ n_c\ n_d\ n_f\ n_k]$. The first four numbers represent the orders of the polynomials of the system and the noise model, and the last number represents the time delay.

The following algorithm is used: First create the angular velocity from the angle by differentiation. Since the measurement noise is fairly small a reasonable estimate of the motor velocity is obtained. This is practical because of the physical nature of the robot includes an integrator, see the physical models in Chapter 3. Then resample with a factor of n=10 (including the low-pass anti alias filter), and remove the mean. Identify with Box-Jenkins $\mathcal{N} = [3\,3\,3\,3\,1]$ and $\mathcal{N} = [5\,5\,5\,5\,1]$. This is done in MATLAB™ code using SITB, (see Ljung, 2000).

The Bode plots of the Box-Jenkins models are shown in Figure 5.3. The model with $\mathcal{N} = [3\,3\,3\,3\,1]$ corresponds to a two-mass flexible model and is called bj3, that is it has the same number of poles. The model with $\mathcal{N} = [5\,5\,5\,5\,1]$ corresponds to a three-mass flexible model (see Section 3.2.3) and is called bj5. The additional resonance peak of bj5 can be seen in Figure 5.3. A comparison is given in Table

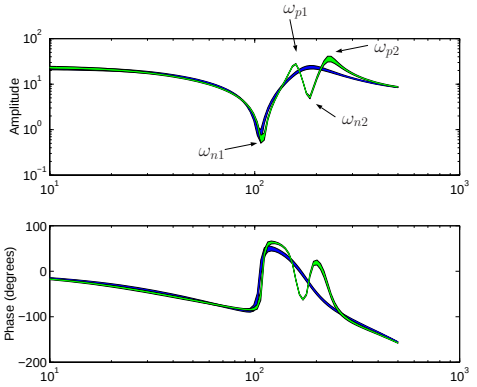


Figure 5.3 *Bode plots of the Box-Jenkins models bj3 (black) and bj5 (grey). They are plotted with 99% confidence intervals. ω_{n1} and ω_{n2} denote the notches and ω_{p1} and ω_{p2} denote the peaks in the Bode plot.*

5.1. The fit is calculated by simulation on the estimation data set. The value of the fit becomes negative when the bj3 model and the entire data set is used in the simulation. The fit is defined as

$$fit\,\% = (1 - \frac{\sum(y(t) - \hat{y}(t))^2}{\sum(y(t) - \bar{y})^2}) * 100 \tag{5.2}$$

A negative fit means that the error is bigger than the output or that the model is worse than guessing $\hat{y}(t) = \bar{y}(t)$. The negative fit of bj3 is a result from that the third order model cannot describe the high frequencies very well. Table 5.1 shows that the fit becomes better if the first 999 data points are not used, which correspond to the high frequency part of the input signal.

	loss	fit	1000:2300	ω_{n1}	ω_{p1}	ω_{n2}	ω_{p2}
bj3	0.20	-12%	78.7%	107	x	x	x
bj5	0.11	78.4%	71.2%	107	156	185	232

Table 5.1 *Fit and frequency comparison of two Box-Jenkins models. The value of the loss function is denoted loss and the fit is simulated fit on the estimation data set. 1000:2300 denotes the simulated fit using the last 1301 data points of the data set, which corresponds to the low frequency part of the data. The frequencies ω_i is defined in Figure 5.3*

5.3 Physically parameterized models

This section describes experiences from identification of physical models. The identification is carried out using SITB in MATLAB™, which uses prediction error minimization and a search algorithm of the type (4.17). A two-mass flexible model is identified in Section 5.3.1. In Section 5.3.2 a three-mass flexible model is identified. The domain of attraction regarding the initial parameter values is studied in Section 5.3.3.

5.3.1 Two-mass flexible model

The models can be described by the following equations with $x(t)$ as states, $u(t)$ as the input signal and $e(t)$ as white noise

$$\begin{aligned} \dot{x}(t) &= A(\theta)x(t) + B(\theta)u(t) + K(\theta)e(t) \\ y(t) &= C(\theta)x(t) + e(t) \end{aligned} \tag{5.3}$$

$A(\theta)$, $B(\theta)$ and $C(\theta)$ are matrices parameterized by the physical parameters $\theta = [J_m\ J_a\ f_m\ k\ d]$ as described in (3.13). K is a column vector which can be included if the noise should be modeled. If the noise is not modeled it will be denoted by $K = 0$. The physical model and the noise model will always share poles if a noise model is used, $K \neq 0$.

First we look at which physical parameters that are reasonable to estimate from input/output data. To answer this question we examine how the Bode plot changes when the different parameters in the physical model change. The state space model with three states given in (3.13) is used. Figure 5.4 shows changes of ±20% in the parameter values. The nominal values of the parameters can be seen in Table 5.2.

J_m	J_a	f_m	k	d
$9 \cdot 10^{-4}$	12	0.1	$1.5 \cdot 10^5$	70

Table 5.2 *Nominal values of the parameters.*

We have the following comments of the Bode plots in Figure 5.4:

1. f_m only affects the low frequency region. Therefore it may be difficult to get a good estimate of f_m from the input signal described in Section 5.1. We may need a dedicated experiment to estimate f_m.
2. The variation of the damping coefficient, d, does not make any evident changes of the Bode plot. Thus, d could be hard to estimate. We can distinguish a small change in the height of the peaks and depth of the notches.
3. k and J_a have similar but opposite effects on the Bode plot. However, the ratio has a significant influence on the place of the notch. This is also illustrated in (5.4).

The angular frequency of the notch in the Bode plot of the physically parameterized model corresponds to the zero in (3.14). It can be calculated by

$$\omega_{n1} = Im\left\{-\frac{d}{2J_a} \pm \sqrt{(\frac{d}{2J_a})^2 - \frac{k}{J_a}}\right\} \approx \sqrt{\frac{k}{J_a}} = 112\, rad/s \tag{5.4}$$

It says that the stiffer the spring is the higher the frequency will be. The depth of the notch is only dependent of $d/(2J_a)$. The smaller this value is the deeper the notch becomes.

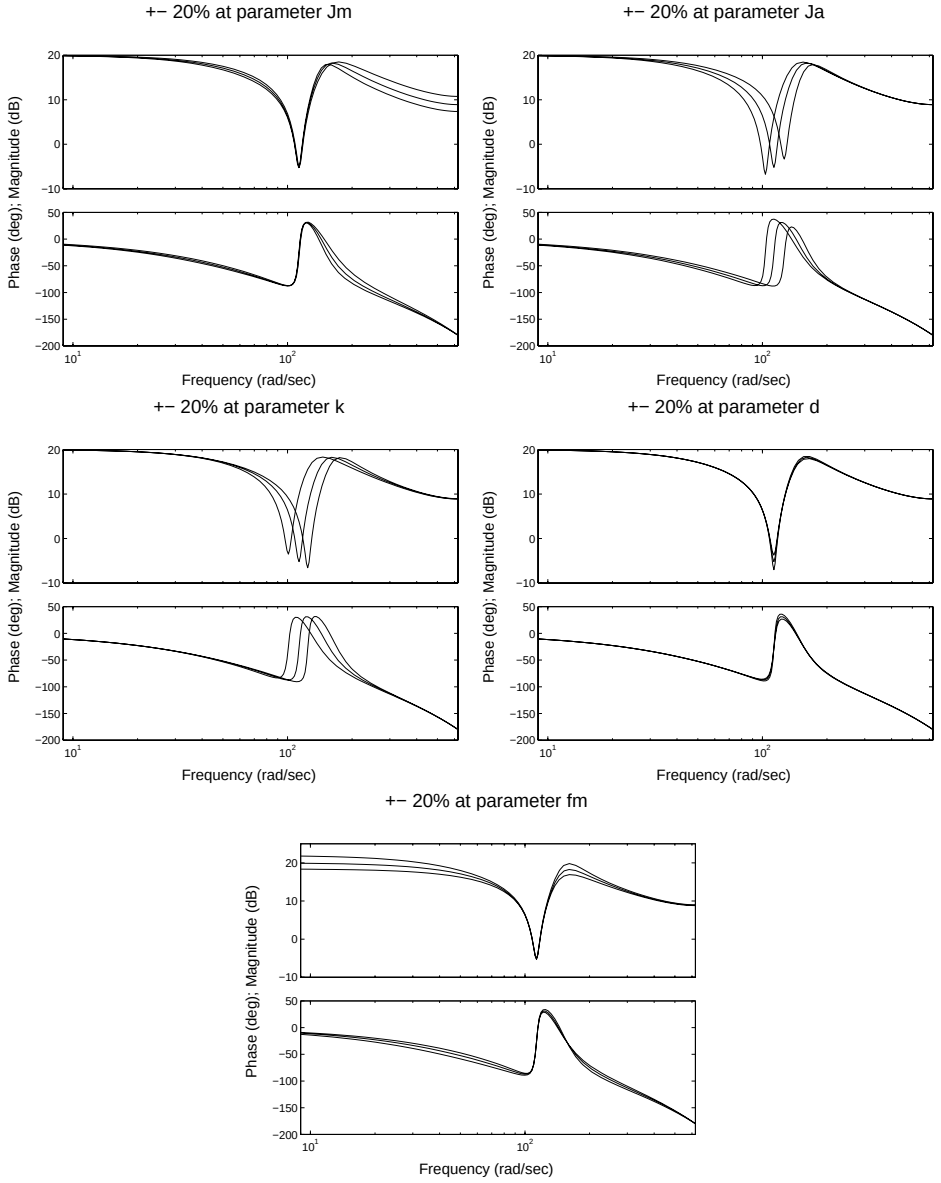


Figure 5.4 *Changes of* $\pm 20\%$ *in parameter values.*

Two-mass flexible model identification

Four cases will be investigated corresponding to identification both with and without damping coefficient, d and noise model, K. The identified models of the four combinations are shown in Figure 5.5. Note that the notch moves to the left when

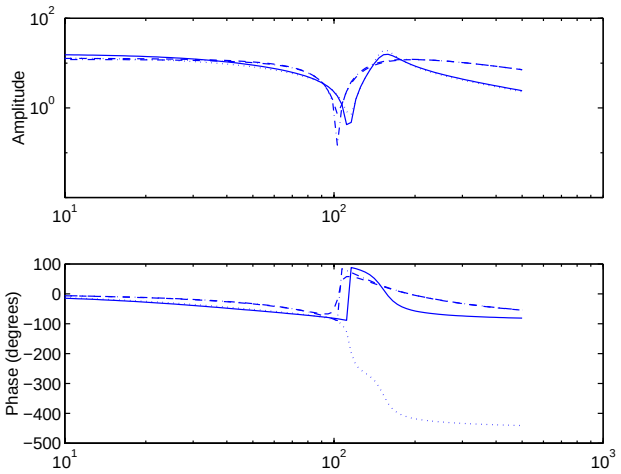


Figure 5.5 *Bode plots of physically parameterized models. Solid: $K = 0$, $d = 0$. Dotted: $K = 0$, $d \neq 0$. Dash dotted: $K \neq 0$, $d = 0$. Dashed: All parameters free ($K \neq 0$, $d \neq 0$).*

the noise model is included in the identification. Table 5.3 shows the frequencies of the notch and the peak. The loss and the simulated fit on the estimation data are also tabulated. The table shows that the loss is drastically decreased when a

	specification	loss	fit	500: 2300	1000: 2300	ω_{n1}	ω_{p1}	n
p1	$K = 0$ $d = 0$	6.21	48.2%	63.7%	69.2%	114	155	4
p2	$K = 0$ $d \neq 0$	5.91	49.4%	64.8%	64.3%	115	157	5
p3	$K \neq 0$ $d = 0$	0.329	37.9%	53.3%	66.0%	103	191[1]	7
p4	$K \neq 0$ $d \neq 0$	0.327	38.2%	52.0%	67.4%	103	197[1]	8

Table 5.3 *Comparison between four physically parameterized models. The fit is simulated fit on the estimation data set. n is the number of estimated parameters. The frequencies ω_i is defined in Figure 5.3.* [1] *These peaks are barely discernible.*

noise model is included ($K \neq 0$), but the simulated fit becomes worse. The loss is defined as the sum of the square of the residuals obtained when the output is compared to the one-step-ahead prediction. Without noise model the prediction is formed entirely using the input signal, i.e, as a simulation of the model. With a model including a noise model the prediction is based on both previous outputs and inputs. It is therefore natural that the loss is drastically decreased when the noise model is included. An example can be that a model which has the predictor equal to the previous output will give a small loss, but is useless for simulation. It also can be seen how the poles are moved to be adjusted to the noise model. A comparison of the physical parameters of the model can be seen in Table 5.4. Note the big difference in J_m if the noise is modeled compared to not modeling the

	J_m (10^{-4})	J_a	f_m (10^{-2})	k (10^5)	d
p1	8.58	10.9	6.30	1.40	0
p2	9.05	11.4	7.55	1.50	-74.4
p3	2.54	8.6	8.26	0.92	0
p4	2.57	8.5	7.81	0.91	41.0

Table 5.4 *Physical parameters of four different models. p1 with $K = 0$, $d = 0$, p2 with $K = 0$, $d \neq 0$, p3 with $K \neq 0$, $d = 0$ and p4 with all parameters free ($K \neq 0$, $d \neq 0$).*

noise.

It is a difficult task to choose which one of these models that is the best. We have chosen p2 because it has the best simulated fit. Its Bode plot, with confidence intervals, can be seen in Figure 5.6. In Figure 5.7 the Bode plots of the black-box models, bj3 and bj5, and the physically parameterized model, p2 are compared.

Some remarks about the identification of the two-mass flexible model:

- The estimates of the parameters are sensitive to different initial parameter values. This is further studied in Section 5.3.3.
- Normalizing the parameters does not make any significant change to the model, although this is recommended for good numerical properties.
- Two different parameterizations has been tried. One using the physical parameters and the other is defined in (5.5). Using the totally changed parameterization (5.5) improves the convergence area, that is the parameter identification is less sensitive to initial values of the parameters. However,

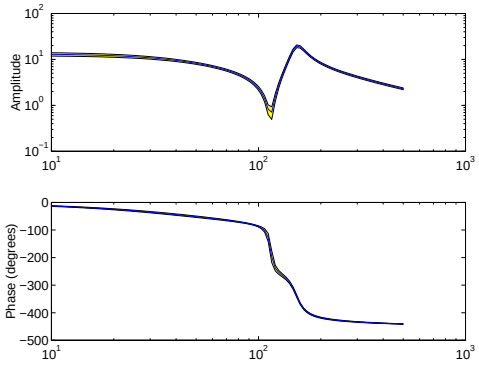


Figure 5.6 *Bode plot of the physically parameterized model p2, where $K = 0$, plotted with 99% confidence interval.*

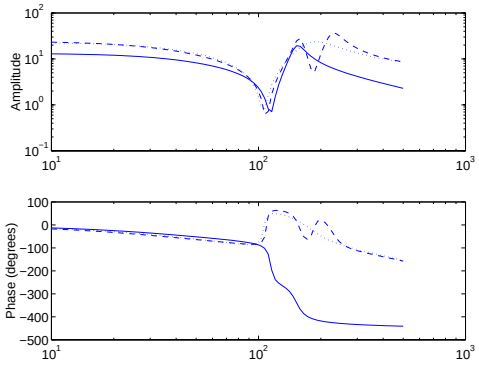


Figure 5.7 *Bode plots of the black-box models bj3 (dotted), bj5 (dashed) and the physically parameterized model p2 (solid).*

any exhaustive evaluation has not been done. For the two-mass model a totally changed parameterization is used:

$$\begin{aligned} J_m &= 1/\theta_3 \\ J_a &= \theta_1/(\theta_2\theta_3) \\ f_m &= \theta_5/\theta_3 - r^2\theta_1\theta_4/(\theta_2\theta_3) \\ k &= \theta_1/\theta_3 \\ d &= \theta_1\theta_4/(\theta_2\theta_3) \end{aligned} \tag{5.5}$$

which gives an A-matrix where most of the parameters appear by themselves:

$$A = \begin{pmatrix} 0 & r & -1 \\ r\theta_1 & \theta_5 & r\theta_1\theta_4/\theta_2 \\ \theta_2 & r\theta_4 & -\theta_4 \end{pmatrix} \tag{5.6}$$

- Often the damping coefficient becomes negative. A possible explanation of this is that the 3 state model tries to resemble the 5 state model (which is a more accurate model class). Because of the two peaks in the five state model the second peak becomes higher and narrower, and this corresponds to a negative d. Another possible explanation is that there exist unmodeled non-linearities that are best approximated with a negative d. Although a negative d is not physical, as long as the friction f_m is high enough the model is stable and can be used for simulation. One can specify that the damper should be positive but this will give a slightly worse model.

5.3.2 Three-mass flexible model

The models can be described by the following equations with $x(t)$ as states, $u(t)$ as the input signal and $e(t)$ as white noise

$$\begin{aligned} \dot{x}(t) &= A(\theta)x(t) + B(\theta)u(t) + K(\theta)e(t) \\ y(t) &= C(\theta)x(t) + e(t) \end{aligned} \tag{5.7}$$

$A(\theta)$, $B(\theta)$ and $C(\theta)$ are matrices parameterized by the physical parameters $\theta = [J_m\ J_g\ J_a\ f_m\ k_g\ k_a\ d_g\ d_a]$ as described in (3.19). K is now a five by one matrix which is included when the noise is modeled.

Four cases will be investigated; with and without damping coefficients, d_g and d_a, and with and without noise model, K. The Bode plots of the identified models of the four combinations are shown in Figure 5.8. They are all very close to each other. A comparison of the fits can be seen in Table 5.6, and in Table 5.5 the frequencies of the notches and peaks are compared. In Table 5.7 the physical parameters are shown.

The values in Table 5.6 show that the models give almost equal fit on the last part of the data. This is because all the models describe the low frequency range fairly well, but they differ if we look at the entire data set, where p6 has the best fit. It is also good if as few parameters as possible are used, and thus p6 is decided to be the "best" model. Its Bode plot, with confidence region, is depicted in Figure 5.9, and in Figure 5.10 it is compared with the black-box models. The Bode plot of the physically parameterized model is close to the Bode plot of the black-box

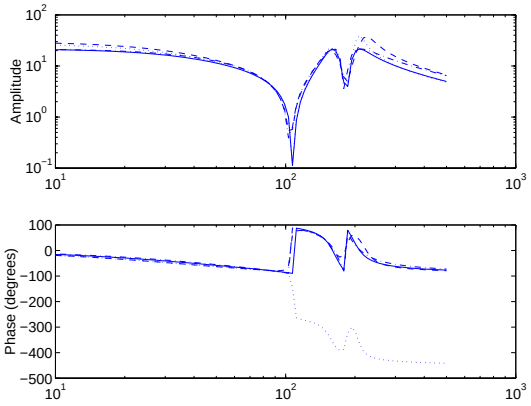


Figure 5.8 *Bode plot of physically parameterized three-mass flexible models. Solid: $K = 0$, $d = 0$. Dotted $K = 0$. Dash dotted: $d = 0$. Dashed: All parameters free.*

	ω_{n1}	ω_{p1}	ω_{n2}	ω_{p2}
p5	107.4	160	183	210
p6	107.3	157	183	209
p7	104.6	163	181	211
p8	105.0	157	184	224

Table 5.5 *Frequency comparison between four physically parameterized three-mass flexible models. The unit of the notches and peaks is [rad/s].*

model bj5. Both models have the same number of poles. One difference is that the black-box model has a noise model, whereas the physically parameterized model has not.

5.3.3 Sensitivity of the initial parameter values

It is important to investigate how sensitive the identification procedure is to variations in the initial parameter values. As a measure we look at how much it is possible to vary the initial value of the parameters one at a time. First the initial value is changed for one parameter, and the initial values of the other parameters are set to the values from the original identification. Then all the parameters are identified. As a criterion that the identification was successful the loss function is compared to the loss function of the original identification. If the new loss function

	specification	loss	fit	1000:2300	n
p5	$K = 0$ $d = 0$	2.30	62.0%	74.5%	6
p6	$K = 0$ $d \neq 0$	1.56	73.4%	70.6%	8
p7	$K \neq 0$ $d = 0$	0.184	58.5%	75.7%	11
p8	$K \neq 0$ $d \neq 0$	0.153	70.9%	70.1%	13

Table 5.6 *Comparison between four physically parameterized models. The fit is simulated fit on the estimation data set. n is the number of estimated parameters.*

	J_m (10^{-4})	J_g	J_a	f_m (10^{-2})
p5	4.37	10.1	1.62	4.62
p6	4.56	9.92	1.72	3.55
p7	3.34	9.51	1.36	4.63
p8	3.39	9.41	2.11	3.17
	k_g (10^5)	k_a (10^5)	d_g	d_a
p5	1.50	0.419	0	0
p6	1.50	0.446	-62	20
p7	1.29	0.357	0	0
p8	1.46	0.506	-39	30

Table 5.7 *Physical parameter comparison between four physically parameterized three-mass flexible models.*

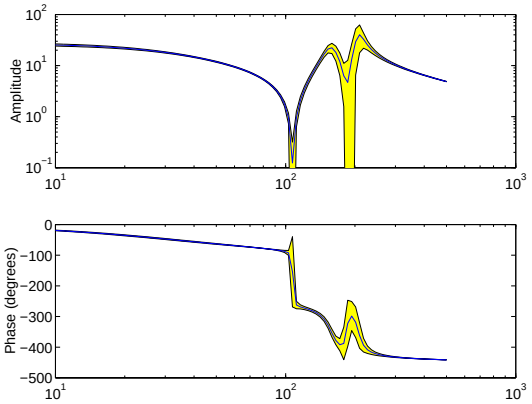


Figure 5.9 *Bode plot of the physically parameterized three-mass flexible model p6, where $K = 0$, plotted with 99% confidence interval.*

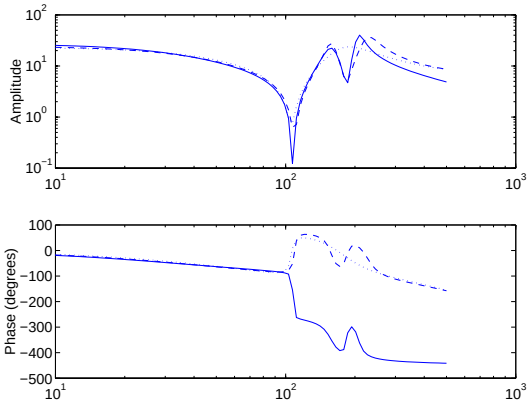


Figure 5.10 *Bode plot of the black-box models bj3 (dotted) and bj5 (dashed). The physically parameterized three-mass flexible model p6 (solid) is also shown.*

is less than 5% above the original loss function the identification is successful. This defines a region shown in Table 5.8, where the initial value of each parameter has an upper and a lower limit, which are calculated using interval halving.

	J_m	J_g	J_a	f_m
upper limit	12200%	253%	585%	10300%
lower limit	40%	68%	52%	32%
	k_g	k_a	d_g	d_a
upper limit	319%	586%	254%	1770%
lower limit	76%	59%	-1530%	-211%

Table 5.8 *Domain of attraction of initial values of the parameters, when varying the initial value of one parameter at a time.*

To investigate what happens when all initial parameters are changed at the same time Monte-Carlo simulations (identifications) are made. The region is defined by taking 20% of the upper limit and lower limit defined by Table 5.8. From this region 1000 identification experiments were made with random initial parameter values. 649 of these identifications were successful, and 351 got more than 5% above the previous loss function. How to find good initial parameter values to the identification algorithm is an ongoing research area see, for example, Xie and Ljung (2002).

5.4 Validation

To check the quality of the models another data set is used. This data set consists of a large number of superimposed sinusoids spanning approximately the same frequencies as the previously used input signal when doing the identification. This frequency range is interesting when doing control for example. A simulated fit can be seen in Figure 5.11. The values of the fit are 51.5% for bj5 and 37.8% for p6

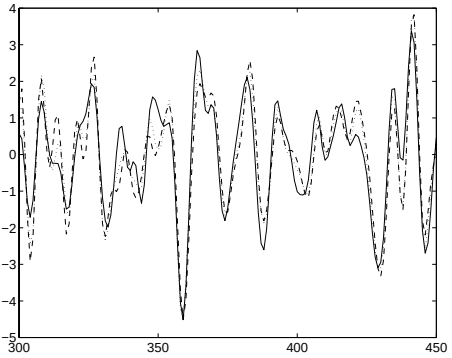


Figure 5.11 *Simulated validation fit of the black-box model bj5 (dotted), the physically parameterized model p6 (dash dotted) and the output (solid) is shown.*

when using (5.2). Note that the physical model has fewer parameters than the black-box model, not counting the noise model. Further cross validation can be found in Östring et al. (2001a).

5.5 Summary

Identification of the dynamics of an ABB IRB 1400 when moving around axis one has been carried out. Discrete time black-box models of Box-Jenkins type as well as well as continuous time physically parameterized models have been tried. Models of order three and and five have been evaluated. One main observation has been that a fifth order model gives a significantly better description of the robot dynamics than a third order model. For the physically parameterized models this implies that a model consisting of three masses gives gives a better description than a two-mass model. The data used for the identification have been collected while the robot is subject to feedback control. Theoretically this implies that a

disturbance description should be included in the model. The identification results do however indicate that a physically parameterized model without disturbance model gives a reasonably good description of the real system. It should be noted that the results presented have been obtained using one data set.

Both black-box models and physically parameterized models can be used for control. Control using a physically parameterized model is further discussed in Chapter 8. Another area where models are used is diagnosis. In diagnosis it is often important to monitor physical aspects of the system. In this thesis off-line identification of physically parameterized models are used as nominal models for diagnosis purposes in Chapter 7.

6

Recursive identification

Off-line identification of physical parameters in continuous time models can be carried out using commercially available software. See, for example, Ljung (2000). In off-line identification a batch of data is used, and typically a criterion function is minimized with respect to the parameter values, see Chapter 5. This is a computationally cumbersome procedure, and the estimates are acquired after a certain time. Recursive identification of the parameter estimates is therefore needed in several areas (Ljung and Söderström, 1983), for example, in adaptive control or monitoring tasks. Recursive identification of parameters in a discrete time model of black-box type is also an established area (Ljung, 1999; Ljung and Söderström, 1983). The topic of this chapter, i.e., to recursively estimate the physical parameters in continuous time models, has however received less attention. This topic is important in cases like fault isolation and fault identification where the main task is to identify the continuous time parameter values as fast as possible after a fault has occurred. It is in this context the recursive identification of physical parameter values comes in. While the problem under consideration is general the attention in this chapter is concentrated to the simplified description of a flexible robot arm.

Continuous time in RLS algorithms is treated in, for example, Huarng and Yeh

(1992) where the model is discrete but the algorithm is in continuous time. In this chapter the reversed is studied; the model is in continuous time while the algorithm is in discrete time. The model used in the identification experiments in this chapter is a three-mass model of a robot arm. This model of the robot arm was introduced in Section 3.2.3.

Section 6.1 gives a short description of the model. In Section 6.2 the recursive identification procedure is discussed, and in Section 6.3 different ways of forming the predictor and its gradient are discussed. The nominal model used in the recursive identification experiments is presented in Section 6.5. Then in Section 6.6 the experiments are shown. A summary is given in Section 6.7.

6.1 The robot system

The dynamics of the robot system (see Figure 5.2) when moving around axis one will be approximated by a model consisting of three masses connected via springs and dampers as shown in Figure 6.1. In Chapter 5 models with two and three masses were compared, and it was found that models with three masses gave considerably better results. Therefore this chapter is restricted to three-mass models. The input is the torque τ generated by the electrical motor, while the output is the motor angle φ_m. The angles of the other masses, φ_g and φ_a respectively, are not measurable (see Section 3.2.3 for details).

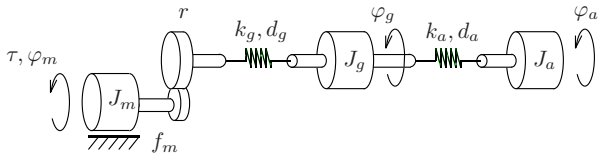


Figure 6.1 *Three-mass flexible model.*

The model is written in state space notation using $u(t) = \tau(t)$ as input and $y(t) = \dot{\varphi}_m(t)$ as output

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + e(t) \end{aligned} \tag{6.1}$$

where the matrices A, B and C are defined in (3.19), and $e(t)$ denotes the measurement noise. In the robot control system the motor angle φ_m is the only available output signal, but since the measurement noise is fairly small a reasonable estimate of the motor velocity is easily obtained. Therefore the motor velocity is used as output signal in the model above.

6.2 The recursive identification algorithm

The identification will be carried out using a recursive prediction error (RPEM) algorithm (Ljung and Söderström, 1983).

$$\hat{\theta}(t) = \hat{\theta}(t-1) + P(t|t)\psi(t)\varepsilon(t) \tag{6.2}$$

where $\varepsilon(t)$ denotes the prediction error

$$\varepsilon(t) = y(t) - \hat{y}(t, \hat{\theta}(t-1)) \tag{6.3}$$

and $\hat{y}(t, \hat{\theta}(t-1))$ is the prediction of the output. $P(t|t)$ is a symmetric (covariance) matrix, and $\psi(t)$ denotes the gradient of the prediction error with respect to the unknown parameters. When designing an appropriate update algorithm for the matrix $P(t|t)$ there are two main issues that have to be taken into account. First, when identifying parameters that are subject to change it is important that the algorithm maintains its tracking ability. Second, it is important to prevent $P(t|t)$ from growing during periods of poor excitation. These issues can be dealt with in different ways, and the aim here is to point out some possible solutions. The update of P is therefore carried out in three steps. The first step, sometimes denoted the measurement update (Anderson and Moore, 1979; Parkum, 1992), is given by

$$P(t|t) = P(t|t-1) - \frac{P(t|t-1)\psi(t)\psi^T(t)P(t|t-1)}{1 + \psi^T(t)P(t|t-1)\psi(t)} \tag{6.4}$$

This updating of P corresponds to the updating the information matrix $R(t|t) = P^{-1}(t|t)$ according to

$$R(t|t) = R(t|t-1) + \psi^T(t)\psi(t) \tag{6.5}$$

In the next step, here denoted the time update, the tracking ability is ensured, and in this chapter two alternatives will be considered. The first alternative is the classical forgetting factor approach, which corresponds to the updating

$$\bar{P}(t+1|t) = \frac{1}{\lambda}P(t|t) \tag{6.6}$$

where $0 < \lambda \leq 1$ denotes the forgetting factor. The second alternative is the Kalman filter approach(Anderson and Moore, 1979), where

$$\bar{P}(t+1|t) = P(t|t) + R_1 \tag{6.7}$$

and R_1 is a symmetric positive definite matrix. In the third step the aim is to ensure that P does not grow without bounds when the excitation is poor. This problem is equivalent to the problem that the information matrix tends to a singular matrix. The standard method for handling this problem is to use regularization

$$R(t+1|t) = \bar{R}(t+1|t) + \mu \cdot I \tag{6.8}$$

where μ is a positive scalar. Using the matrix inversion lemma this corresponds to

$$P(t+1|t) = \bar{P}(t+1|t)(I + \mu\bar{P}(t+1|t))^{-1} \tag{6.9}$$

i.e., the regularization of the information matrix corresponds to a normalization of the covariance matrix. See also Gunnarsson (1996). The design parameters are hence λ and R_1 for the tracking properties and μ for the regularization. Different aspects of the choice of these parameters will be discussed in connection to the experiments later in this chapter.

In the experiments later in this chapter the excitation is good, and regularization is not needed. To summarize the first algorithm without the normalization (setting $\mu = 0$ in (6.9)), (6.4), (6.6) and (6.9) are combined using the notation $P(t) = P(t|t)$

$$P(t) = \frac{1}{\lambda}\left[P(t-1) - \frac{P(t-1)\psi(t)\psi^T(t)P(t-1)}{\lambda + \psi^T(t)P(t-1)\psi(t)}\right] \tag{6.10}$$

From (6.2), $P(t|t)\psi(t)$ can also be written as

$$K(t) = \frac{P(t-1)\psi(t)}{\lambda + \psi^T(t)P(t-1)\psi(t)} \tag{6.11}$$

which also can be used in (6.10). The algorithm based on forgetting factor, λ, is

Algorithm 6.1 (RPEM using forgetting factor)

$$\varepsilon(t) = y(t) - \hat{y}(t, \hat{\theta}(t-1)) \tag{6.12}$$

$$P(t) = \frac{1}{\lambda}\left[P(t-1) - \frac{P(t-1)\psi(t)\psi^T(t)P(t-1)}{\lambda + \psi^T(t)P(t-1)\psi(t)}\right] \tag{6.13}$$

$$K(t) = \frac{P(t-1)\psi(t)}{\lambda + \psi^T(t)P(t-1)\psi(t)} \tag{6.14}$$

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\varepsilon(t) \tag{6.15}$$

The second alternative is inspired by the Kalman filter approach (see (6.7)) and based upon the assumption that the parameters are time-varying like a random walk:

$$\theta(t+1) = \theta(t) + v(t) \tag{6.16}$$

where $Ee^2(t) = R_2$ and $Ev(t)v^T(t) = R_1$. For the algorithm properties it is only the relation between R_1 and R_2 that is important. Therefore R_2 can be normalized to 1. Note that if R_2 is set to one $P(t)$ is not the covariance of $\tilde{\theta} = \theta - \hat{\theta}(t)$. To get the covariance $P(t)$ must be scaled with R_2. To summarize this second alternative, after

some calculations using (6.2), (6.4), and (6.7) and denoting $P(t) = R_2 P(t|t-1)$ the following algorithm is acquired and also found in text books (e.g., Ljung, 1999; Gustafsson, 2000):

Algorithm 6.2 (RPEM based on Kalman filter)

$$\varepsilon(t) = y(t) - \hat{y}(t, \hat{\theta}(t-1)) \tag{6.17}$$

$$P(t) = P(t-1) - \frac{P(t-1)\psi(t)\psi^T(t)P(t-1)}{R_2 + \psi^T(t)P(t-1)\psi(t)} + R_1 \tag{6.18}$$

$$K(t) = \frac{P(t-1)\psi(t)}{R_2 + \psi^T(t)P(t-1)\psi(t)} \tag{6.19}$$

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\varepsilon(t) \tag{6.20}$$

6.3 Forming the predictor and its gradient

A key point when applying the RPEM algorithms above is how to determine $\hat{y}(t, \theta)$ and $\psi(t)$ for the continuous time model when only discrete time data are available. The continuous time state space model in (6.1) can be converted to discrete time using standard methods assuming zero order hold of the input. However, the resulting matrices of the discrete time state space model are complicated functions of the physical parameters, and this makes the differentiation complicated. Another way is to transform the continuous time model to discrete time approximately with, for example, Euler's approximation.

Here two other ways of calculating the predictor, and its gradient are investigated. The first approach taken here is to carry out the operations in the reversed order, i.e., to do the differentiation using the continuous time model, and in a second step convert the expression for the gradient to discrete time. This is presented in Section 6.3.1. The second approach is to transform the continuous time model to discrete time, and then calculate the gradient using numerical differentiation. This is presented in Section 6.3.2.

6.3.1 Forming the predictor and its gradient in continuous time

Consider a linear state space model for which the predictor is

$$\begin{aligned} \dot{x}(t, \theta) &= A(\theta)x(t, \theta) + B(\theta)u(t) \\ \hat{y}(t, \theta) &= C(\theta)x(t, \theta) \end{aligned} \tag{6.21}$$

The predictor is chosen in this way because off-line identification experiments give good results using OE models. The gradient of the prediction with respect to a scalar parameter is given by, see, for exmaple, Ljung and Glad (1994),

$$\psi^{(i)}(t) = \frac{d}{d\theta^{(i)}}\hat{y}(t,\theta) = C(\theta)z(t) + \bar{C}^{(i)}(\theta)x(t,\theta) \tag{6.22}$$

where $\theta^{(i)}$ denotes the ith component of θ and

$$z^{(i)}(t) = \frac{d}{d\theta^{(i)}}x(t,\theta) \qquad \bar{C}^{(i)}(\theta) = \frac{d}{d\theta^{(i)}}C(\theta) \tag{6.23}$$

The time derivative of $z^{(i)}(t)$ is obtained from

$$\frac{d}{dt}z^{(i)}(t) = \frac{d}{d\theta^{(i)}}\dot{x}(t,\theta) = A(\theta)z^{(i)}(t) + \bar{A}^{(i)}(\theta)x(t,\theta) + \bar{B}^{(i)}(\theta)u(t) \tag{6.24}$$

where

$$\bar{A}^{(i)}(\theta) = \frac{d}{d\theta^{(i)}}A(\theta) \qquad \bar{B}^{(i)}(\theta) = \frac{d}{d\theta^{(i)}}B(\theta) \tag{6.25}$$

Introducing the extended state vector

$$X(t) = (z^{(i)}(t) \;\; x(t))^T \tag{6.26}$$

the state space description for the prediction, and the gradient is given by

$$\begin{aligned} \dot{X}(t) &= \begin{pmatrix} A(\theta) & \bar{A}^{(i)}(\theta) \\ 0 & A(\theta) \end{pmatrix} X(t) + \begin{pmatrix} \bar{B}^{(i)}(\theta) \\ B(\theta) \end{pmatrix} u(t) \\ \begin{pmatrix} \psi(t) \\ \hat{y}(t) \end{pmatrix} &= \begin{pmatrix} C(\theta) & \bar{C}^{(i)}(\theta) \\ 0 & C(\theta) \end{pmatrix} X(t) \end{aligned} \tag{6.27}$$

Equations (6.23) and (6.25) are repeated for each parameter that should be identified. This means that for each parameter that is identified the state space model in (6.27) is extended with n more states. Depending on which parameter that is of interest the matrices $\bar{A}^{(i)}(\theta), \bar{B}^{(i)}(\theta)$ and $\bar{C}^{(i)}(\theta)$ will have different properties. Considering, for example, identification of f_m in (6.27) one gets

$$\bar{A}^{(i)}(\theta) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{J_m} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad \bar{B}^{(i)}(\theta) = 0 \qquad \bar{C}^{(i)}(\theta) = 0 \tag{6.28}$$

Then (6.27) is transformed to discrete time using the current estimate $\hat{\theta}(t)$ and assuming zero order hold.

6.3.2 Forming the predictor and its gradient using numerical differentiation

An alternative to Section 6.3.1 is to calculate the discrete time model, $G(q,\theta)$, from the continuous time counterpart numerically each time it is needed, and then perform numerical differentiation. The state space model is transformed to its discrete time counterpart in each sampling point, using the current parameter estimate, $\hat{\theta}(t)$. Let the continuous time state space model be given by

$$\begin{aligned} \dot{x}(t) &= A_c(\theta)x(t) + B_c(\theta)u(t) \\ y(t) &= C_c(\theta)x(t) \end{aligned} \tag{6.29}$$

The identification will be carried out using sampled data. Assuming zero order hold the model is given in discrete time by the state space model

$$\begin{aligned} x((k+1)T) &= F(\theta)x(kT) + G(\theta)u(kT) \\ y(kT) &= H(\theta)x(kT) \end{aligned} \tag{6.30}$$

where

$$F(\theta) = e^{A_c(\theta)T}, \quad G(\theta) = \int_0^T e^{A_c(\theta)t}B_c(\theta)\,dt, \quad H(\theta) = C_c(\theta) \tag{6.31}$$

are solved numerically for the current value of $\hat{\theta}(t)$ in every sample.

The model can be seen as an output error (OE) like model structure, and the predictor can be written

$$\begin{aligned} \hat{y}(t,\theta(t-1)) = -a_1\hat{y}(t-1,\theta(t-1)) - \ldots - a_{n_a}\hat{y}(t-n_a,\theta(t-1)) \\ +b_1u(t-1) + \ldots + b_{n_b}u(t-n_b) \end{aligned} \tag{6.32}$$

To be able to search for the optimum the gradient of $\hat{y}(t,\theta)$ w.r.t θ must be calculated or estimated in order to find a search direction. Therefore a numerical approximation of the gradient $\psi^T(t,\theta) = d/d\theta(\hat{y}(t,\theta))$ is used

$$\psi^{(i)}(t,\theta^{(i)}) = \frac{\hat{y}(t,\theta^{(i)}(t-1) + \Delta/2) - \hat{y}(t,\theta^{(i)}(t-1) - \Delta/2)}{\Delta} \tag{6.33}$$

where Δ is a design variable.

Evaluation

In off-line identification experiments OE-models give good results. Therefore it would be preferable that the recursive identification algorithm also uses OE like models. Simulations have shown that the way of computing the predictor and its

gradient in Section 6.3.2 sometimes diverges for the OE like model. On the other hand, forming the predictor and its gradient according to Section 6.3.1, which also uses an OE like model, gives good results. Therefore the method in Section 6.3.1 is chosen for the recursive identification experiments later in this chapter.

Here we try to indicate why the OE like structure has problems in Section 6.3.2 using numerical differentiation when computing the predictor and its gradient. Start with the OE model

$$y(t,\theta) = \frac{B(q,\theta)}{A(q,\theta)}u(t) + e(t) \tag{6.34}$$

The predictor becomes

$$\hat{y}(t,\theta) = (1 - A(q,\theta))\hat{y}(t,\theta) + B(q,\theta)u(t) \tag{6.35}$$

The calculation of $\psi(t)$ is studied

$$\psi^T(t) = \frac{d}{d\theta}\hat{y}(t,\theta) \tag{6.36}$$

This derivative is calculated using a difference approximation

$$\psi^{(i)}(t) \approx \frac{\hat{y}(t,\theta^{(i)} + \Delta/2) - \hat{y}(t,\theta^{(i)} - \Delta/2)}{\Delta} \tag{6.37}$$

This becomes, using the OE like model structure

$$\begin{aligned}\psi(t) \approx & \frac{\left[B(q,\theta^{(i)} + \Delta/2) - B(q,\theta^{(i)} - \Delta/2\right]u(t)}{\Delta} \\ & + \frac{(1 - A(q,\theta^{(i)} + \Delta/2))\hat{y}(t,\theta^{(i)} + \Delta/2)}{\Delta} \\ & - \frac{(1 - A(q,\theta^{(i)} - \Delta/2))\hat{y}(t,\theta^{(i)} - \Delta/2)}{\Delta}\end{aligned} \tag{6.38}$$

The last terms include

$$[\hat{y}(t-1,\theta^{(i)} + \Delta/2) \;\; \hat{y}(t-2,\theta^{(i)} + \Delta/2) \;\; \cdots \;\; \hat{y}(t-n,\theta^{(i)} + \Delta/2)] \tag{6.39}$$

and

$$[\hat{y}(t-1,\theta^{(i)} - \Delta/2) \;\; \hat{y}(t-2,\theta^{(i)} - \Delta/2) \;\; \cdots \;\; \hat{y}(t-n,\theta^{(i)} - \Delta/2)] \tag{6.40}$$

The problem is that these terms are not available. To get them the system must be simulated from $t = 0$. Therefore both these must be approximated using

$$[\hat{y}(t-1,\theta(t-1)) \;\; \hat{y}(t-2,\theta(t-2)) \;\; \cdots \;\; \hat{y}(t-n,\theta(t-n))] \tag{6.41}$$

This approximation may give a bad estimate of ψ. To get a better estimate of ψ it is possible to simulate a few steps back in time. Below the problem of using these approximations is shown in an example.

Example 6.1 ***$\psi(t)$ calculation***
The system (a first order system, $A(q,\theta) = 1 - \alpha q^{-1}$ and $B = \theta$)

$$y(t) = \frac{\theta}{1 - \alpha q^{-1}} u(t) + e(t), \tag{6.42}$$

where θ is the gain, has the predictor

$$\hat{y}(t,\theta) = \theta u(t) + \alpha \hat{y}(t-1,\theta) \tag{6.43}$$

The gradient with respect to θ is

$$\psi(t,\theta) = u(t) + \alpha \psi(t-1,\theta) \tag{6.44}$$

This is compared to using the difference approximation given by (6.38). Then the gradient becomes instead

$$\psi(t) = \frac{\Delta u(t)}{\Delta} = u(t) \tag{6.45}$$

which is not the same as (6.44).

6.4 Some properties of the gradient

An important property when identifying physical parameters is that the character of the gradient $\psi(t)$ depends on which particular parameter that is identified. Consider the transfer operator of the model converted to discrete time

$$y(t) = G(q,\theta)u(t) + e(t) \tag{6.46}$$

where $e(t)$ is white noise, since, as mentioned above, an output error structure is considered. The prediction is hence given by

$$\hat{y}(t,\theta) = G(q,\theta)u(t) \tag{6.47}$$

The gradient of the prediction with respect to a scalar $\theta^{(i)}$ can be expressed as

$$\psi(t) = G_{\theta^{(i)}}(q,\theta)u(t) \tag{6.48}$$

where $G_{\theta^{(i)}}(q,\theta)$ is the derivative of the transfer operator $G(q,\theta)$ with respect to $\theta^{(i)}$. The variance of $\psi(t)$ will have a big influence on the properties of the estimates, and it can be expressed

$$\bar{E}[\psi^2(t)] = \frac{1}{2\pi} \int_{-\pi/T}^{\pi/T} \left| G_{\theta^{(i)}}(e^{i\omega T},\theta) \right|^2 \Phi_u(\omega) d\omega \tag{6.49}$$

The magnitude of the variance hence depends on the character of the input spectrum and the properties of the transfer function $G_{\theta^{(i)}}(q,\theta)$. This will be illustrated in a the recursive identification experiments below.

6.5 Identification of nominal model

First, off-line identification is used to get the parameter values of the model. The acquired parameter values are used as nominal values in the recursive algorithm. Then some of the parameters are recursively estimated.

The determination of the nominal model is done by identifying a physically parameterized model as described in Chapter 5. The external excitation that is added to the reference signal of the robot control system is a sum of sinusoids in the range $0 - 60\pi$ rad/s. The signal is created by setting the discrete Fourier transform of the reference signal to one with random phase between $0 - 60\pi$ rad/s, and then transforming the Fourier transform to time domain, i.e, the following frequencies are excited

$$\frac{2\pi k}{12} \text{ rad/s}, \qquad k = 1, \ldots, 360 \tag{6.50}$$

This is very similar to filtered white noise, but a difference is that equal energy in the specified frequency points is acquired. This type of signals is further investigated in Pintelon and Schoukens (2001); Norrlöf et al. (2002). The system used for collecting data from the robot is further described in Norrlöf (2000). The sampling frequency of the data is 200 Hz. Since the system is operating in closed loop the applied torque signal will be affected by the feedback. The properties of the torque signal are shown in Figure 6.2, and it is seen that the input energy is low below 70 rad/s with peaks around 95, 125 and 185 rad/s.

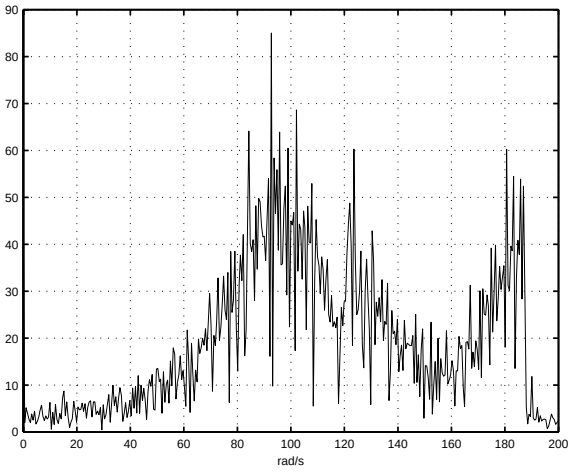


Figure 6.2 *FFT of the torque signal.*

The System Identification Toolbox in MATLAB™ Ljung (2000) is used in the off-

line identification. For comparison a very high order ARX model is shown together with the acquired model in Figure 6.3. This is used because a high order ARX model is capable of approximating any linear system arbitrary well (Ljung, 1999, page 336). As seen in the figure the three-mass model is a reasonable approximation

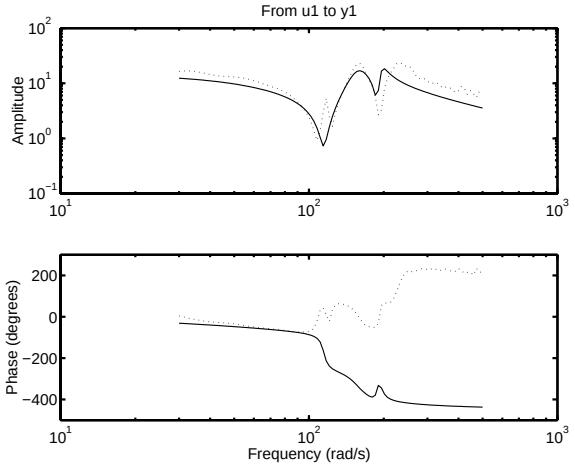


Figure 6.3 *Bode plot of the physical model (solid) and a high order ARX model (dotted).*

of the system. The acquired parameter values from the off-line identification will be used as nominal parameter values in the recursive identification below.

6.6 Recursive identification experiments

In this section the recursive identification experiments are presented. Before the results in Section 6.6.3 are given a discussion on which parameters that should be recursively identified is provided in Section 6.6.1, and the design variables are studied in Section 6.6.2.

6.6.1 Identified parameters

The choice of which parameters that should be identified recursively originates from which parameter values that are likely to change over time. An example could be a worn gear box, which indicates an increase in f_m. In these experiments the interest has been concentrated on the three parameters k_T, J_a^{-1}, and f_m, where k_T denotes

the static gain in the system. Why these parameters are chosen is further discussed in Chapter 7. To give an indication of that the particular parameter to be identified will have influence on the algorithm behavior the squared amplitude curve of the transfer operators $G_{\theta^{(i)}}(q,\theta)$ is studied. Figure 6.4 shows $\left|G_{\theta^{(i)}}(e^{i\omega T},\theta)\right|^2$ for the three parameters considered here where the parameter values are set to the nominal parameter values from the off-line identification. This figure together with (6.49) and Figure 6.2 indicate that the variance of $\psi(t)$ will be highest for f_m, less for J_a^{-1} and lowest for k_T.

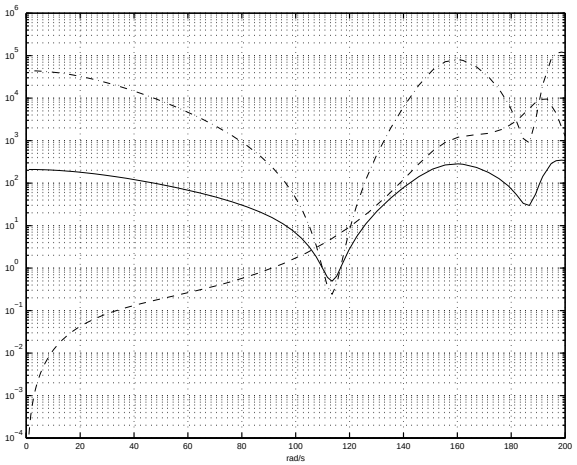


Figure 6.4 *Plot of* $\left|G_{\theta^{(i)}}(e^{i\omega T},\theta)\right|^2$ *for the three parameters. Solid line:* k_T*. Dashed line:* J_a^{-1}*. Dash-dotted line:* f_m*.*

6.6.2 Design variables

The recursive identification will be carried out using both the forgetting factor and the covariance matrix modification versions of the RPEM algorithm. The design parameters to choose are the initial values of θ and P respectively and also the forgetting factor λ and the matrix R_1. The choices of λ and R_1 are trade offs between tracking ability of the algorithm and the variance of the parameter estimates. In this application $\lambda = 0.995$ has been found to be an appropriate value. Since the input can be chosen almost freely it can be designed to give sufficient excitation to avoid windup problems for the matrix P. Hence the regularization parameter can be put to zero in these experiments.

The choice of $P(0)$ can be made from different viewpoints. In case the algorithm is

going to be used with some kind of change detection it is realistic to assume that $P(0)$ can be chosen to give a fast convergence after sudden changes in the true parameters. Without change detection the tracking properties will be determined by the current values of P, which depend on the choice of λ and the properties of the input signal via $\psi(t)$. Assuming $\psi(t)$ to be quasi-stationary, see Ljung (1999), and λ to be close to one the matrix P can in steady state, see Ljung and Gunnarsson (1990), be approximated by $\mathbf{P}$ given by

$$\mathbf{P} = (1 - \lambda)Q^{-1} \tag{6.51}$$

and

$$\mathbf{P}Q\mathbf{P} = R_1 R_2 \tag{6.52}$$

respectively, where

$$Q = \lim_{N\to\infty} \frac{1}{N} \sum_{t=1}^{N} \psi(t)\psi^T(t) \tag{6.53}$$

As discussed above the variance of the components of $\psi(t)$ will be of different magnitude, and hence also the elements of $\mathbf{P}$ will be of different magnitude. This will then give different tracking and variance properties of the different parameter estimates. Using forgetting factor there is only one design variable available to affect the trade off between tracking and variance. Using the covariance modification the matrix R_1 offers more freedom for dealing with this problem.

The aim here is to show the algorithm properties without any change detection involved. In order to select an initial value of P that represents the steady state behavior the update equation for P is first run using the nominal parameter values in the computation of $\psi(t)$. The mean value of the diagonal elements in P are then used to form the initial value $P(0)$ in the actual identification. The initial value of the parameters in θ is set to 20% above the nominal values of the parameters to see how well the parameters adapt.

6.6.3 Results

The parameter adaptation for the forgetting factor case is shown in Figure 6.5, and it can be seen that all parameters converge to their nominal values. The convergence rate is approximately the same for the estimates of J_a^{-1} and f_m respectively, while it is somewhat higher for the estimate of k_T. In Figure 6.6 the diagonal elements of P are shown. The big difference in magnitude is caused by the big difference in the magnitude of the elements in $\psi(t)$, see Figure 6.4.

An identification experiment is also carried out where the covariance matrix modification is used. The aim in this experiment is only to illustrate that this freedom

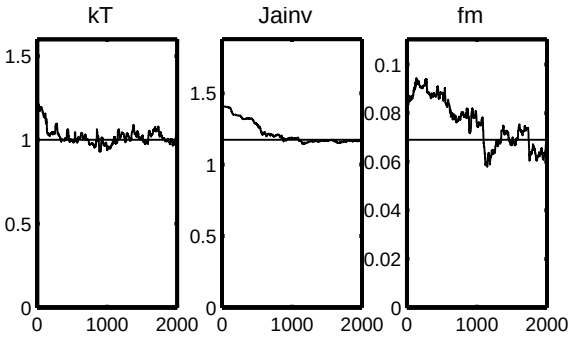


Figure 6.5 *Parameter estimates using forgetting factor.*

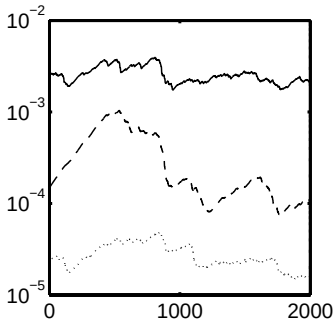


Figure 6.6 *Diagonal elements of P. Solid: P_{11}. Dashed: P_{22}. Dotted: P_{33}.*

is available. How to use it in a suitable way is left for further work. Assume, for example, that it is desired that the diagonal elements of P are of the same order of magnitude. Due to the difference in magnitude in the elements of $\psi(t)$ it is then necessary to let the elements of R_1 be of different magnitude. In the experiments the choice

$$R_1 = \rho \begin{pmatrix} 5 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 500 \end{pmatrix} \tag{6.54}$$

is used, where $\rho = 10^{-8}$. Figures 6.7 and 6.8 show the results from this experiment. Figure 6.8 shows that the diagonal elements of P now have the same magnitude. The changes in algorithm properties are illustrated in Figure 6.7. The convergence rate for k_T is slower, and the variance is lower than for the forgetting factor case. The variance of the estimate of f_m is much higher due to the increase in the corresponding element in P. The results indicate that it is possible to handle the

tracking and variance trade off more or less individually for the different parameters by suitable choices of the elements in R_1.

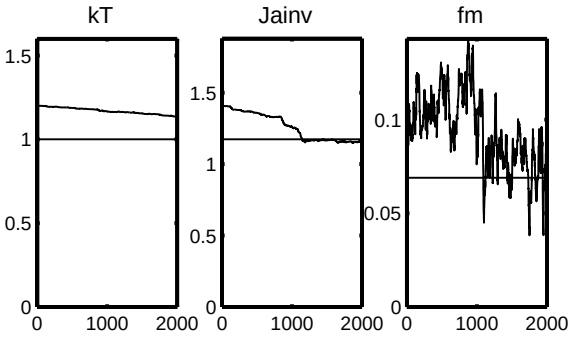


Figure 6.7 *Parameter estimates using covariance matrix modification.*

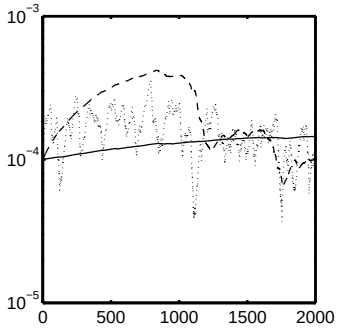


Figure 6.8 *Diagonal elements of P. Solid: P_{11}. Dashed: P_{22}. Dotted: P_{33}.*

In these experiments the excitation could be chosen almost freely. In practice it can not be expected that the input signal is sufficiently exciting all the time during real operation. How to deal with this problem using, for example, the regularization procedure presented above is left for further work.

6.7 Summary

Recursive identification of the dynamics of an ABB IRB 1400 when moving around axis one has been carried out. The model structure is continuous time, physically parameterized, and based on a three-mass approximation of the true dynamics. In

the first stage an off-line identification is carried out in order to obtain nominal values of the physical parameters. In the second stage some of the parameters are identified recursively using a second data set recorded from the robot system. A recursive prediction error minimization algorithm is used. The gradient of the predictor is obtained by first differentiating the continuous time model with respect to the unknown parameters, and then converting the problem to discrete time. Both the forgetting factor and the covariance matrix modification approaches for achieving tracking capability are tried. The results indicate that the covariance matrix modification (Kalman filter) approach offers a greater freedom to deal with the tracking ability and disturbance rejection trade off.

Areas where recursive identification is used are, for example, adaptive control and diagnosis. In this thesis diagnosis is further discussed in Chapter 7, where recursive estimation of physical parameters are used as a monitoring tool.

7

Diagnosis

There are several reasons for applying diagnosis tools to a system. Some of those reasons are listed below:

- **Safety**. Faults can affect the safety of the system. A good example is an aircraft where it is extremely important to detect minor faults before they grow so big that the aircraft may malfunction.
- **Maintenance**. In many cases the diagnosis system is helpful when it comes to maintenance. It can be useful in pointing out parts which should be replaced. It is important to reduce the number of maintenance stops of a plant. Sometimes a diagnosis system can also help to decide the time when the plant should be stopped for maintenance.
- **Machine protection**. In order to protect a machine against further damage diagnosis can be applied to detect parts that probably will fail soon, and thus protect the machine from further damage by doing a controlled stop.
- **Performance**. Often a diagnosis system can detect faults which are not fatal for the intended task, but affects the performance of the system.

There are of course many more examples where diagnosis can be used. Here diagnosis methods will be applied to an industrial robot using the models and algorithms from previous chapters with mainly the performance and the maintenance as objectives.

First the basic concepts are introduced in Section 7.1. Then the focus will be more on the industrial robot application, and the possible faults are discussed in Section 7.2. In Section 7.3 fault detection based on parameter estimation is studied. Two ways of doing fault detection using parameter estimation are further studied in Section 7.4, and the results are shown for two of these methods in Section 7.5 using real data. In Section 7.6 a summary is given.

7.1 Basic concepts

This section presents some of the basic concepts in the area of diagnosis. The terminology is becoming more unified, but there are still many different terminologies originating from different approaches. Here the following will be used:

- **Fault**
  Some characteristic property that deviates from normal or usual behavior.
- **Fault detection**
  To determine if a fault has occurred. Sometimes this also includes to determine the time of the fault. Here it will only be used to determine if a fault has occurred.
- **Fault isolation**
  To determine which component that is faulty.
- **Fault identification**
  To determine important properties of the fault, for example, the magnitude of the fault.
- **Fault diagnosis**
  Here there are three major views. One view is that fault detection and isolation is referred to as diagnosis (Gustafsson, 2000). Another view is that fault diagnosis includes both isolation and identification (Gertler, 1998). There are also authors who use fault diagnosis for all three properties; detection, isolation, and identification. When it is important the words detection, isolation, and identification are used instead of diagnosis to make the distinction clear.
- **Diagnosis**
  A diagnosis system produces diagnoses. A diagnosis is a possible explanation

of which faults or faulty components that can explain the behavior of the system.

- **FDI**
  An acronym for Fault Detection and Isolation.
- **Residual**
  Signal used in the diagnosis system. It is used for different things in different areas, and it is usually determined by the context. Often it is some signal that is close to zero in the fault free case and differs significantly from zero if a fault has occurred. Other words for similar signals are test variable or test quantity.

Another fundamental concept in the area of diagnosis is redundancy. Redundancy means that it is possible to calculate a variable or signal in at least two different ways. These signals can then be compared, and if they differ there is a fault in the system. Redundancy can be obtained by using redundant sensors (physical redundancy), or by looking at a plant behavior and compare it to what is expected from the basis of a mathematical model (analytical redundancy).

Different methods apply to different kinds of faults. Therefore one usually distinguishes between different types of faults. A fault can belong to several categories:

- **Additive faults**
  These are faults modeled by unknown inputs to the system. In a fault free case these inputs are normally zero, but in case of a fault the inputs cause a change in the plant outputs. Typical examples of this type of faults are leaks in pipes or bias in a sensor.
- **Multiplicative faults**
  These are faults that can be modeled by a change in some of the parameters in the model of the system. These parameter changes affect the outputs of the system depending on the size of the inputs.
- **Abrupt changes**
  A fault can appear abruptly. A parameter or signal changes from one value to another at one time instant.
- **Incipient changes**
  This is when a fault appears gradually. Typically this happens when machinery gets worn leading to deteriorating performance of the machine.
- **Intermittent changes**
  Faults that appear and then disappear repeatedly.

The circumstances under which the diagnosis system is working are also important. Active diagnosis is distinguished from passive diagnosis. In active diagnosis a

dedicated experiment is performed to detect and isolate the fault. It is therefore possible to choose the input signals so that as many faults as possible can be observed. This in contrast to passive diagnosis when the diagnosis system is run during normal operation of the plant. This usually means that the algorithm must be run on-line, and that it is not possible to chose the input signal. In some cases the data are collected from normal operation, but the diagnosis algorithm is run off-line.

Robustness is also an important issue in fault diagnosis. In a real system there are different operating points, the measurements may be noisy, and the models are always subject to modeling errors (Frank and Ding, 1997). It will be further investigated later in this chapter when the fault detection algorithms are presented.

Fault diagnosis is a large area. An introduction to the subject can be found in Isermann (1997). In the literature there has been a wide interest in fault diagnosis over the last two decades. The early work was surveyed by Isermann (1984) and in the book of Patton et al. (1989). A later survey was made by Isermann and Ballé (1997). The model-based fault detection methods can be categorized in several somewhat overlapping approaches:

1. Parity equations approach (Gertler, 1998). This is rearranged input and output relations, subject to a linear transformation. The freedom of choosing the transformation can be used for disturbance decoupling and fault isolation. Disturbance decoupling means that the disturbance does not affect the residual or test variable.
2. Diagnostic observers (Frank and Ding, 1997). An observer is constructed for monitoring the plant.
3. Parameter estimation (Isermann, 1997). Parameter estimation methods are used to estimate parameters which describe certain faults. Parameter estimation is a central part of system identification which is throughly described in Ljung (1999); Söderström and Stoica (1989); Unbehauen and Rao (1987).

These are all methods related to the area of control. There are several other methods originating from other areas such as multivariate statistics, neural networks, geometric distance, fuzzy logic and signal processing such as band-pass filters and FFT methods. There are also knowledge-based methods and methods from the AI perspective. In Chiang et al. (2001) several of these methods are described.

7.2 The industrial robot application

In this thesis an industrial ABB robot is studied. The robot is described in detail in Chapter 2. To begin with it is important to look at which faults that are likely to happen:

- Error in the torque constant, which is a static gain in the inner control loop of the motor current.
- Torque disturbance on the motor side. This can, for example, originate from increased friction in the bearings or revolution dependent ripple.
- Torque disturbance in the gear box caused by increased friction, bigger backlash, or torque pulsations from the gears.
- Torque disturbance on the arm side. This can be caused by collisions, loose cables, or torques from tools. It can also originate from errors in the load parameters when the operator has not given the correct load parameters for the tool or workpiece.
- Additive disturbance on the measured motor position. This disturbance can originate from bad cables, electrical disturbance from the power line or a faulty resolver.

Here the focus is on the faults that can be described by a parameter change. In the following three faults have been chosen. These are

1. Incorrect torque constant. This results in a change in parameter k_T.
2. Increased viscous friction in the motor. This affects the parameter f_m.
3. Incorrect load parameters. The affected parameter is the inertia of the arm, J_a.

Parameter estimation methods will be applied to detect and isolate these faults in this chapter. The primary concern is the isolation of the faults, since this is important in this application.

7.3 Parameter estimation methods for fault diagnosis

Early work within parameter estimation methods for fault detection and isolation was surveyed by Isermann (1984). In Gertler (1998) a semi-batch algorithm is suggested for on-line parameter estimation (ordinary sliding window), which can

be made semi-recursive. Gertler (1998) uses black-box discrete time models for the estimation and then calculates the physical counterpart of the parameters in a second step via iterations. This approach has difficulties handling the problem of only estimating some of the continuous time parameters. This is often the case when only a few of the physical parameters are monitored. One way of dealing with this problem is direct identification of continuous time models. In Dixon et al. (2000) parameter estimation methods are used for decoupling of parameters in a two arm robot when estimating a free swinging joint and locked joint faults. Other examples using parameter estimation for fault diagnosis can be found in Moseler and Iserman (1998) and Broussard and Trahan (1991).

Often the diagnosis system produces residuals sensitive to different faults. A check if the residual differs significantly from zero is made in a second step. It is important that the residual or the check is insensitive to noise and different excitations of the system. To change the residual or the check according to these objectives is called robustification.

Robustification

Common for all diagnosis algorithms using parameter estimation is the need for some kind of robustification against noise and different input signals. To make the algorithms robust often adaptive thresholds based on variance of the residual are used to determine if the residual differs significantly from zero. An equivalent way of doing this is to scale the residual and have a constant threshold, which is called normalization. More information on robustness issues can be found in Chen and Patton (1999).

Another way of making the algorithms more robust is the use of the CUSUM algorithm (Gustafsson, 2000). This replaces the usual threshold. Let the residual be denoted $\theta_i(t)$. The CUSUM algorithm can be written as

$$\begin{aligned} g(t) &= g(t-1) + \theta_i(t) - \nu \\ g(t) &= 0, \text{if } g(t) < 0 \\ g(t) &= 0, \text{and } t_a = t \text{ and alarm if } g(t) > h \end{aligned} \tag{7.1}$$

The drift variable ν can be based on an adaptive threshold, and the threshold h is a measure of how long time the threshold ν must be violated. Note that if the residual can be negative another CUSUM test must be run in parallel using $-\theta_i(t)$ as input. A successful example using a CUSUM test can be found in Bøgh (1995).

There are other more or less ad hoc methods that can be adopted, for example, the alarm is only generated if the residual is greater than the threshold for more

than ten of the last 20 samples.

7.4 Two ways of doing fault diagnosis based on parameter estimation

In this section two methods of fault detection and isolation using parameter estimation methods are studied.

7.4.1 The classical approach

In this approach the process parameters are monitored. Much of the work in this area is inspired by Isermann (1984). It is also close to the validation methods in the identification area (Ljung, 1999; Söderström and Stoica, 1989). The test variable used is the deviation of parameter values from the nominal parameter values

$$T_i(t) = \hat{\theta}_i(t) - \theta_{0,i} \tag{7.2}$$

where the nominal parameter value $\theta_{0,i}$ is estimated from fault free data. One test variable is used for each possible fault. If the test variable is larger than a threshold, an alarm is generated. The parameter vector $\theta_i(t)$ can be estimated using a recursive estimation algorithm estimating only the possible fault parameters.

Robustification can be achieved using an adaptive threshold. A common choice is to scale the threshold using an estimate of the standard deviation at each time point.

$$\left|\hat{\theta}_i(t) - \theta_{0,i}\right| < 3\sigma_i(t) \tag{7.3}$$

If this is violated an alarm (detection) is generated. The test variable that exceeds the threshold indicates which parameter that is faulty. Fault identification is also achieved if the algorithm is run for some more time. An estimate of the standard deviation can either come from the recursive estimation algorithm or be estimated from $\hat{\theta}_i(t)$.

Instead of $\sigma_i(t)$ in (7.3) it is possible to include the variance of the parameter estimated using non faulty data. Then (7.3) becomes

$$\left|\hat{\theta}_i(t) - \theta_{0,i}\right| < 3\sqrt{\sigma_i(t)^2 + \sigma_{0,i}^2} \tag{7.4}$$

where $\sigma_{0,i}$ is the standard deviation of $\theta_{0,i}$

For even more robustification the CUSUM algorithm (7.1) can be used together with the adaptive threshold by setting ν in the CUSUM algorithm to $\nu = 3\sigma_i(t)$.

7.4.2 An approach based on hypothesis test and decision structure

This approach is described in Nyberg (1999). In this approach several separate identifications according to different hypotheses are performed. There is a no fault hypothesis called NF and a fault hypothesis called Fi, where i stands for the specific fault i. The test quantities are defined as

$$\text{NF: } T_0 = \sum_{t=1}^{N} (y(t) - \hat{y}(t, \theta_0))^2 \tag{7.5}$$

$$\text{Fi: } T_i = \min_{\theta_i} \sum_{t=1}^{N} (y(t) - \hat{y}(t, \theta_i))^2 \tag{7.6}$$

When a test quantity exceeds its threshold the hypothesis representing the test quantity is rejected. These test quantities are only valid for one fault at a time, but the method can be extended to multiple faults as well. To evaluate these test quantities a decision structure can be used. Below is an example of three parameters explaining one fault each.

	T_0	T_1	T_2	T_3
F1	X	0	X	X
F2	X	X	0	X
F3	X	X	X	0
NF	0	0	0	0

The interpretation of a 0 is that the test quantity in that column is not affected by a fault in that row. An X means that the test quantity may be affected when a fault in that row is present, i.e., that the value of the test quantity is above the threshold. Typically for very small parameter faults the threshold is not violated, and therefore the X is needed. When adding other methods for other types of faults it is important to have this kind of decision structure based on some influence structure to make it easier to make the correct decisions.

7.5 Results

The experiments begin with investigating the classical approach with several different ways of estimating the adaptive threshold and ways of filtering the parameter estimates in order to get a more robust fault detection and isolation algorithm. Then the approach based on hypothesis tests is studied.

The objective is to detect and isolate as small faults as possible, and at the same time get few false alarms and reduce the missed detection rate as far as possible. This is of course dependent on the size of the parameter faults. It will also be seen that different faults have different difficulties when it comes to detection and isolation, depending on how they influence the system.

In the test setup parameter faults are simulated using real data. This is done by changing the initial values of the parameters in the recursive estimation algorithm, and then look at how the algorithm estimates the parameter values.

The nominal parameters are estimated using the off-line identification procedure described in Chapter 5. In the diagnosis algorithm, using RPEM, only those parameters that correspond to faults are estimated. All other parameter values are set to the nominal values acquired from the off-line algorithm. As initial value these nominal parameter values are chosen for the fault free parameters. To simulate a fault the initial value of the faulty parameter is changed. In the plots in this section only the deviations from these initial values are shown, and if the recursive parameter estimate is clearly separated from zero there is a fault. In this way it is possible to simulate faults using real data that are abrupt (step-like) at time $t = 0$. Note that the off-line identification of the nominal parameters and the diagnosis experiment is run on two different data sets covering the same spectrum.

7.5.1 The classical approach

An experiment, using classical approach described in Section 7.4.1, is shown in Figure 7.1. In this experiment the input is chosen to excite all the parameters in the model. This is the same type of input as used in Chapter 6. In the first row a 20% fault in k_T is simulated. Together with the estimate of the parameter $\hat{\theta}(t)$ an adaptive threshold is shown. The threshold is chosen as

$$h_i(t) = 3\hat{\sigma}_i(t) \tag{7.7}$$

which is an estimate of the standard deviation calculated from $P(t)$ in the recursive algorithm (6.10)

$$\hat{\sigma}_i(t) = \sqrt{\lambda_0 P^{(i,i)}(t)} \tag{7.8}$$

where λ_0 is the variance of the noise. In these experiments this is estimated to approximately 1. It can be estimated recursively using the innovations or be seen as a design parameter. The corresponding test variable is

$$T_i(t) = \hat{\theta}_i(t) - \theta_{i,0} \tag{7.9}$$

where $\theta_{i,0}$ is the nominal parameter value of that parameter. (Note: To be more accurate the adaptive threshold should be modified with the variance of the nominal

parameter value from the off-line algorithm, $\sqrt{\hat{\sigma}_i(t)^2 + \sigma_{i,0}^2}$. Here $\sigma_{i,0}$ is so small that it is approximated with zero.) The diagnosis algorithm clearly detects the

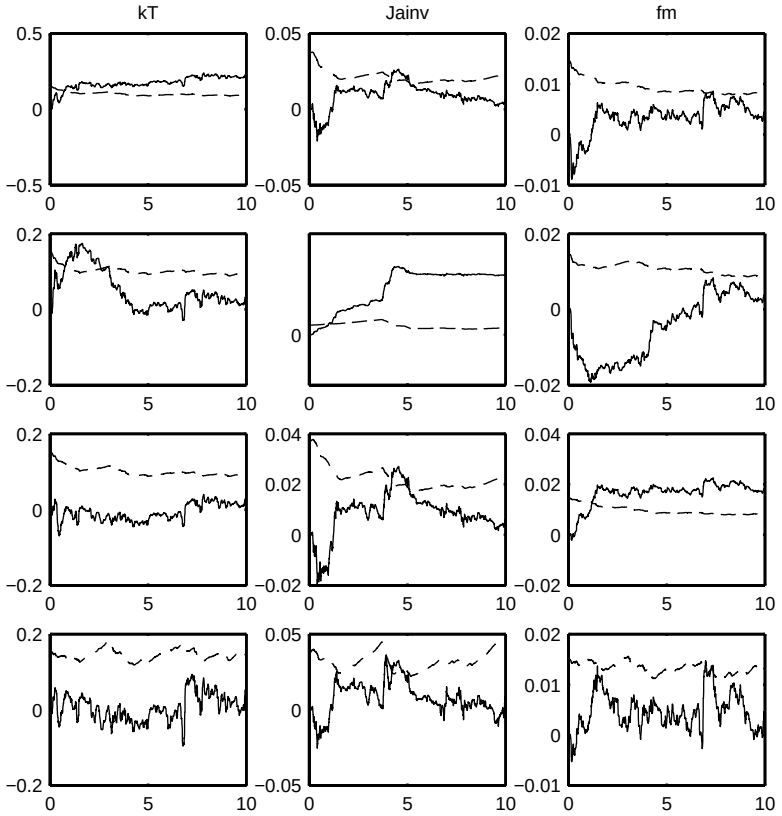


Figure 7.1 *Fault detection and isolation based on recursive parameter estimation. The deviation from the nominal parameter estimates (solid) and the adaptive threshold (dashed) based on $\hat{\sigma}_i(t)$ is shown. In the first row there is a 20% change in k_T, in the second row 20% change in J_a^{-1} and in the third row a 20% change in f_m. In the last row there is no fault present.*

faults which are the upper diagonal plots in Figure 7.1. On the other hand there are also other parameters that signal alarms with the current threshold. This indicates several faults at the same time which is not the case. In the last row there are also two false alarms. There is therefore a need for further robustification.

Robustification using a new estimate of $\hat{\sigma}_i(t)$

To make the algorithm more robust a way of estimating the variance $\hat{\sigma}(t)$ and a smoothed estimate of the parameter estimates $\bar{\theta}(t)$ is tested. The smoothed estimate of $\hat{\theta}(t)$ is calculated as

$$\bar{\theta}(t) = \frac{1}{\sum_{k=1}^{t} \lambda^{t-k}} \sum_{k=1}^{t} \lambda^{t-k} \hat{\theta}(k) \tag{7.10}$$

If it is assumed that the algorithm has run for a long time it is possible to look at

$$\bar{\theta}(t) = \frac{1}{\sum_{k=-\infty}^{t} \lambda^{t-k}} \sum_{k=-\infty}^{t} \lambda^{t-k} \hat{\theta}(k) = (1-\lambda) \sum_{k=-\infty}^{t} \lambda^{t-k} \hat{\theta}(k) \tag{7.11}$$

which can be implemented in a recursive way

$$\bar{\theta}(t) = \lambda \bar{\theta}(t-1) + (1-\lambda)\hat{\theta}(t) \tag{7.12}$$

In the same manner it is possible to estimate the variance recursively

$$\hat{\sigma}_i^2(t) = \lambda \hat{\sigma}_i^2(t-1) + (1-\lambda)(\hat{\theta}_i(t) - \bar{\theta}_i(t))^2 \tag{7.13}$$

and use the smoothed estimate in the test variable

$$T_i(t) = \bar{\theta}_i(t) - \theta_{i,0} \tag{7.14}$$

and the estimated $\hat{\sigma}_i(t)$ in the adaptive threshold. The results are shown in Figure 7.2. When these are used there are no missed detections and the fault isolation is correct after some time. In the last row there is no fault and no fault detected (no false alarms).

Robustification using a CUSUM test

A CUSUM test is applied using the parameter estimates. In order to make the threshold adaptive, ν is chosen to vary with time. The design parameters in (7.1) are chosen as

$$\nu(t) = \alpha_1 \hat{\sigma}_i(t) \tag{7.15}$$

$$\hat{\sigma}_i(t) = \sqrt{\lambda_0 P^{(i,i)}(t)} \tag{7.16}$$

$$h = \alpha_2 \theta_{0,i} \tag{7.17}$$

First the test variable $g(t)$ is only allowed to have a positive drift if the parameter value is more then $\alpha_1 \hat{\sigma}_i(t)$. Then the threshold should be dependent on the size of

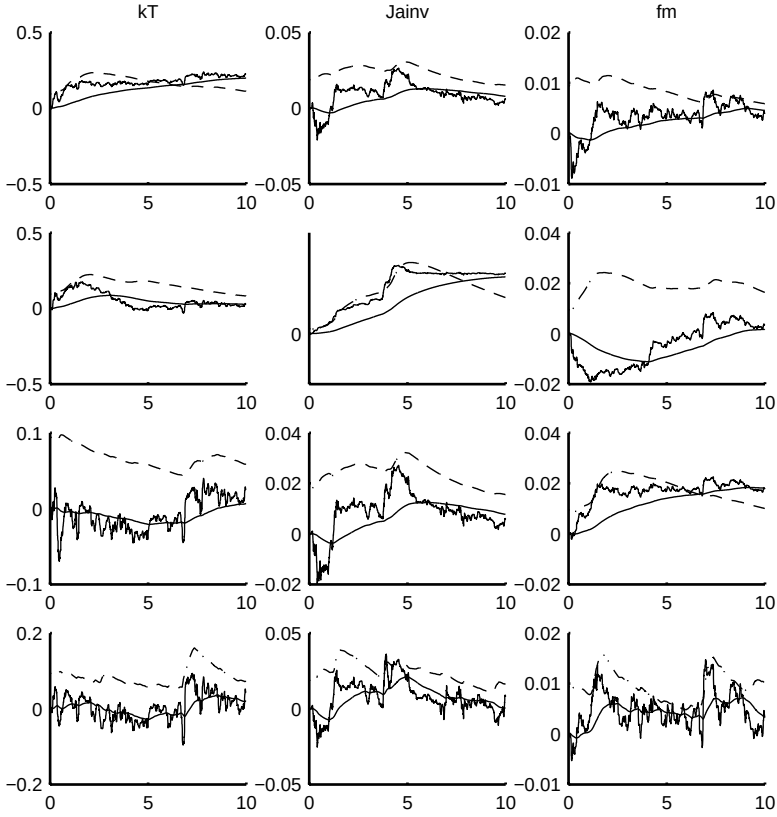


Figure 7.2 *Parameter based fault detection and isolation. Solid lines: Deviation from nominal parameter estimates using the RPEM algorithm, $\hat{\theta}_i(t)$ and the smoothed estimate of the parameter, $\bar{\theta}_i(t)$. Dashed: Adaptive threshold $h(t) = 3\hat{\sigma}_i(t)$ using (7.12). In the first row there is a 20% change in k_T, in the second row 20% change in J_a^{-1} and in the third row a 20% change in f_m. In the last row there is no fault present.*

the parameter, and it is therefore chosen to be proportional to the nominal value of the parameter. In the experiment the design parameters are chosen as $\alpha_1 = 3$ and $\alpha_2 = 20$. Figure 7.3 shows the CUSUM test variable without resetting it when an alarm is present.

One drawback when using more robustification algorithms is that more design parameters must be chosen. Not only the forgetting factor of the recursive estimation

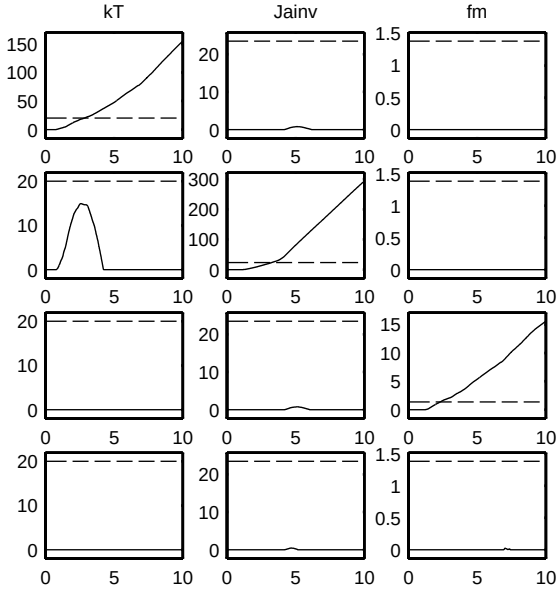


Figure 7.3 *A CUSUM test applied to parameters estimates in Figure 7.1. The test variable, $g(t)$, (solid) is shown together with the constant threshold (dashed). The first row has a fault in k_T, the second a fault in J_a^{-1} and the third a fault in f_m. In the last row there is no fault present.*

algorithm, but also the drift $\nu(t)$ and threshold h. This must be done for every fault and every time point. Here a suggestion is to select the time dependency of the design parameters according to (7.15) and (7.17), leaving only a few scalar design parameters to be tuned.

7.5.2 The approach based on hypothesis tests and decision structure

In this example fault detection based on hypothesis tests and decision structure is investigated, see Section 7.4.2. The input to the system is the same as in the previous experiments. Here the intention is to make the algorithm recursive and use recursive parameter estimation. In order to make the algorithm recursive the

sums are changed to

$$\text{NF: } T_{NF}(t) = \sum_{k=1}^{t} (y(k) - \hat{y}(k, \theta_0))^2 \tag{7.18}$$

$$\text{Fi: } T_i(t) = \sum_{k=1}^{t} (y(k) - \hat{y}(k, \hat{\theta}_i(k)))^2 \tag{7.19}$$

where $\hat{\theta}_i(t)$ is acquired using a recursive estimation of only one parameter. These test variables are then normalized using a recursive estimate of $\hat{\theta}(t)$, which denotes the recursive estimation of all the possible fault parameters. The normalization is then calculated as

$$\text{NF: } T_{NF}(t) = \frac{\sum_{k=1}^{t} (y(k) - \hat{y}(k, \theta_0))^2}{\sum_{k=1}^{t} (y(k) - \hat{y}(k, \hat{\theta}(k)))^2} \tag{7.20}$$

$$\text{Fi: } T_i(t) = \frac{\sum_{k=1}^{t} (y(k) - \hat{y}(k, \hat{\theta}_i(k)))^2}{\sum_{k=1}^{t} (y(k) - \hat{y}(k, \hat{\theta}(k)))^2} \tag{7.21}$$

By using the hypothesis thinking the test variables were defined. Then the test variables were modified to be used on-line (recursive) and also normalized. A decision structure according to the hypothesis tests is used to make the correct decisions. The decision structure is chosen as

	T_{NF}	T_{k_T}	$T_{J_a^{-1}}$	T_{f_m}
F1	X	0	X	X
F2	X	X	0	X
F3	X	X	X	0
NF	0	X	X	X

where the last row has got additional X:s compared to the one in Section 7.4.2, because of the recursive parameter estimation and the normalization. This means that when the test variable for no fault, T_{NF}, is zero the mode no fault is always included in the diagnosis statement. This also assumes that there is only one fault at a time. The results are shown in Figure 7.4. The threshold has been chosen to one, which is a natural choice after normalization. In the first few seconds there is a transient behavior. If the algorithm is used on line the time window is chosen to a constant, not as here where the widow size is equal to the time. This setup should only be used for evaluation. From the test variable $T_{Ja^{-1}}$ in the second row, it can be seen that the window size must be rather large to make the correct decision.

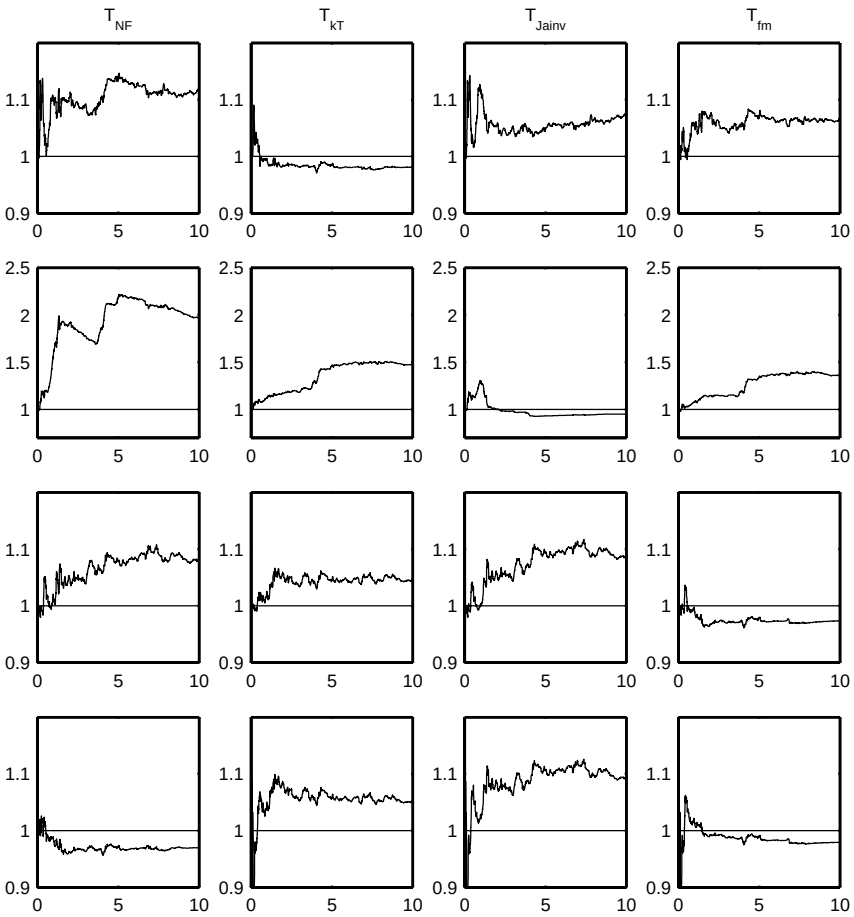


Figure 7.4 *Fault detection and isolation based on (7.20) and (7.21). The first row has a fault in k_T, the second a fault in J_a^{-1} and the third a fault in f_m. In the last row there is no fault present.*

After five seconds the following test variables are violating the thresholds

	T_{NF}	T_{k_T}	$T_{J_a^{-1}}$	T_{f_m}	S
F_{k_T}	1	0	1	1	$\{F_{k_T}\}$
$F_{Ja^{-1}}$	1	1	0	1	$\{F_{J_a^{-1}}\}$
F_{f_m}	1	1	1	0	$\{F_{f_m}\}$
NF	0	1	1	0	$\{NF,\ f_m\}$

where S is the diagnosis statement acquired using the decision structure previously defined. From the first row the hypotheses that there is no fault, the parameter J_a^{-1} is faulty or the parameter f_m is faulty can all be rejected, leaving only k_T as a possible fault explanation, which is correct. The same is true for the second and third row. In the last row using the current threshold it is not possible to separate the no fault hypothesis from a fault in f_m.

7.6 Summary

Diagnosis of three different parametric faults have been carried out using data from an ABB IRB 1400. The diagnosis was based on recursive identification of relevant parameters in a continuous time physically parameterized three-mass model. In a first stage a nominal model was obtained using off-line identification. These parameter values were used to represent the fault free case. In the next stage the estimates from the recursive identification algorithm where used to form different test quantities. For robustification smoothed parameter estimates, adaptive thresholds, and CUSUM tests were tried.

In order to compare the methods the preconditions must be more equal. Our experience is that the test variables used in the hypothesis based approach need a longer time horizon before a correct decision is made, but the hypothesis way of thinking can be a good way of dealing with the results from the classical approach. The approach will probably perform better for off-line or batch wise evaluation. One novel suggestion is to use the parameter estimate at time t and evaluate for a batch of data.

In this chapter we have chosen the input. It is desirable to run the algorithm during a normal robot movement. This is left for future work. The fault detection and isolation does not take into account the correlations between the variables, and ways of using that information would be interesting to study. We have shown two conceptually different ways of doing fault detection and isolation. It is possible to take different parts form these algorithms and form a new way of doing diagnosis based on parameter estimation. An example is to use the threshold in (7.13) together with the parameter estimate and a CUSUM test. Then deal with the results using a decision structure, which is good if other diagnosis methods are added. This is left for further studies.

8

LQG control for disturbance rejection

Control of robot arms is often considered to be a typical servo problem, where the main task is to make the tool follow a given trajectory. It is, however, also important to have good rejection of load disturbances acting on the system, and this is particularly important for robots containing flexibilities. The aim of this chapter is to investigate what LQG control can offer for obtaining good disturbance rejection properties. This choice of control method is a logical second step after conventional PID-control since the controlled variable (arm angle) is different from the measured variable (motor angle). It is then natural to use a control method that utilizes an observer to estimate the controlled variable. The discussion in this chapter is originally based on Östring (1998). It is further treated in Östring and Gunnarsson (1999); Gunnarsson and Östring (2001).

This chapter is organized as follows. In Section 8.1 the system to be studied is presented briefly. Section 8.2 contains a formulation of the control problem, and some relevant transfer functions are derived. This section also contains a brief summary of the LQG design method. In Section 8.3 a numerical example is presented, where it is illustrated that good disturbance rejection properties can be obtained, but that these properties are achieved using an unstable regulator. The following sections

treat various aspects of the use of unstable regulators, concerning, for example, handling of input saturation and robustness. In Section 8.7 the use of feedback from arm acceleration is studied, and it is illustrated that good disturbance rejection properties can be obtained also using a stable regulator, and in Section 8.8 a summary is given.

8.1 System description

The study in this chapter will be based on the two-mass model, introduced in Chapter 3, shown in Figure 8.1.

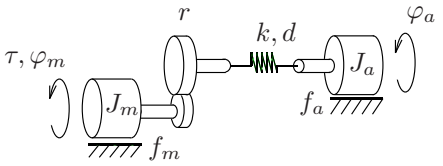


Figure 8.1 *Two-mass model.*

Compared to the model in Section 3.2.2 here a slightly different setup of state variables is used

$$x_1 = \varphi_m \quad x_2 = \dot{\varphi}_m \quad x_3 = \varphi_a \quad x_4 = \dot{\varphi}_a \tag{8.1}$$

where φ_m denotes the angle of the first mass, and φ_a is the angle of the second mass. The gear box ratio r is set to 1, which can be achieved by scaling the arm inertia. There are also additional disturbances added. Let $w(t)$ and $v(t)$ represent load disturbances acting on the first and second mass respectively. Then the state space model is described by

$$\dot{x}(t) = Ax(t) + Bu(t) + B_w w(t) + B_v v(t) \tag{8.2}$$

$$y(t) = Cx(t) \tag{8.3}$$

where u denotes the applied torque, $y = \varphi_m$, and

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{k}{J_m} & -\frac{d+f_m}{J_m} & \frac{k}{J_m} & \frac{d}{J_m} \\ 0 & 0 & 0 & 1 \\ \frac{k}{J_a} & \frac{d}{J_a} & -\frac{k}{J_a} & -\frac{d+f_a}{J_a} \end{pmatrix} \tag{8.4}$$

$$B = \begin{pmatrix} 0 \\ \frac{1}{J_m} \\ 0 \\ 0 \end{pmatrix} \quad B_w = \begin{pmatrix} 0 \\ \frac{1}{J_m} \\ 0 \\ 0 \end{pmatrix} \quad B_v = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{J_a} \end{pmatrix} \tag{8.5}$$

$$C = (1 \quad 0 \quad 0 \quad 0) \tag{8.6}$$

The parameters J_m and J_a denote the moment of inertia of each mass while f_m and f_a denote the viscous friction coefficient of each mass. Finally k and d denote the stiffness and damping between the two masses.

An important point in the first part of this chapter is the assumption that only φ_m is available. The main goal is of course to control φ_a in an appropriate way but the feedback has to be based on measurements of φ_m. This differs from the majority of publications dealing with flexible systems where normally φ_a is available. These assumptions are made in order to describe the typical situation in robot control where the motor angle is measured while the arm angle is the controlled variable.

An obvious extension is to consider the use of additional sensors, and in Section 8.7 the use of feedback from an accelerometer measuring the arm angular acceleration is studied.

8.2 Control

Consider now the two-mass model controlled by the two degrees of freedom regulator as shown in Figure 8.2.

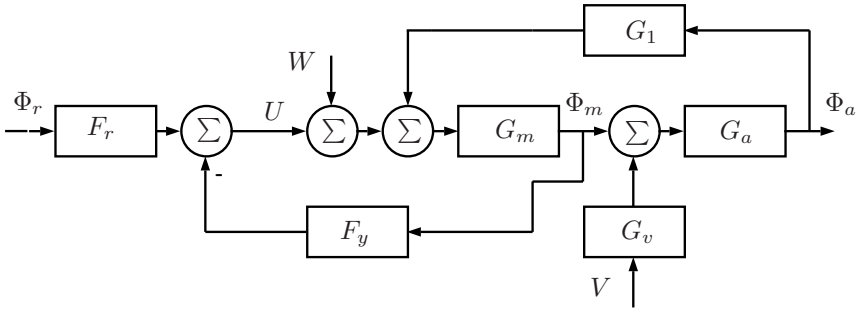


Figure 8.2 *Two-mass model controlled by a two degrees of freedom regulator.*

There are a number of different transfer functions of interest when investigating the properties of the closed loop system, namely from r, w and v to φ_m and φ_a. The relationship between the input signals and the motor angle is

$$\Phi_m(s) = G_{rm}(s)\Phi_r(s) + G_{wm}W(s) + G_{vm}(s)V(s) \tag{8.7}$$

where

$$G_{rm}(s) = \frac{F_r(s)G_{um}(s)}{1 + F_y(s)G_{um}(s)} \quad G_{vm}(s) = G_a(s)G_{um}(s)S(s) \tag{8.8}$$

$$G_{wm}(s) = G_{um}(s)S(s) \tag{8.9}$$

$$S(s) = \frac{1}{1 + F_y(s)G_{um}(s)} \tag{8.10}$$

and $G_{um}(s)$ in the transfer function from u to φ_m (see (8.28) for details). Using

$$\Phi_a(s) = G_a(s)(\Phi_m(s) + G_v(s)V(s)) \tag{8.11}$$

gives

$$\Phi_a(s) = G_{ra}(s)\Phi_r(s) + G_{wa}W(s) + G_{va}(s)V(s) \tag{8.12}$$

where

$$G_{ra}(s) = G_{rm}(s)G_a(s) \quad G_{wa}(s) = G_a(s)G_{um}(s)S(s) \tag{8.13}$$

and

$$G_{va}(s) = G_a(s)(G_{vm}(s) + G_v(s)) \tag{8.14}$$

The servo properties captured in $G_{ra}(s)$ are determined by both $F_r(s)$ and $F_y(s)$, and they can, in principle, be chosen arbitrarily by choosing $F_r(s)$ in an appropriate way. There are of course practical limitations, like input power limitations and model uncertainty, that have to be taken into account. The load disturbance rejection properties, captured in $S(s)$, are entirely determined by $F_y(s)$.

The aim here is to give a brief summary of design of LQG regulators. A thorough presentation can be found in, for example, Friedland (1986). The system is controlled using feedback from estimated states

$$u(t) = -L\hat{x}(t) + l_0 r(t) \tag{8.15}$$

where the state estimate is obtained from the Kalman filter

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + K(y(t) - C\hat{x}(t)) \tag{8.16}$$

The gain vector L is chosen by minimizing the integral

$$\int_0^\infty x^T(t)Q_1x(t) + u^T(t)Q_2u(t)dt \tag{8.17}$$

where Q_1 and Q_2 are appropriately chosen weight matrices. The gain in the Kalman filter is obtained by minimizing the covariance matrix of the estimation error $\tilde{x}(t) = x(t) - \hat{x}(t)$. The state disturbances are then modeled as stochastic processes with covariance matrix R_1. It is also assumed that the measured signal $y(t)$ is affected by a stochastic measurement disturbance $e(t)$ with covariance matrix R_2.

The feedback given by equation (8.15) with the state estimate generated by (8.16) can be described using transfer functions as

$$U(s) = F_r(s)R(s) - F_y(s)Y(s) \tag{8.18}$$

where the transfer functions $F_r(s)$ and $F_y(s)$ are given by ($l_0 = 1$)

$$F_r(s) = 1 - L(sI - A + BL + KC)^{-1}B \tag{8.19}$$

and

$$F_y(s) = L(sI - A + BL + KC)^{-1}K \tag{8.20}$$

Furthermore

$$G_C(s) = C(sI - A + BL)^{-1}B \tag{8.21}$$

and

$$S(s) = 1 - G_C(s)L(sI - A + KC)^{-1}K \tag{8.22}$$

It is seen that the choice of L determines the servo properties, while the disturbance rejection properties are determined by both L and K. Given that L has been fixed by the servo requirements the disturbance rejection properties are determined by K. The order of the transfer functions $F_r(s)$ and $F_y(s)$ resulting from the LQG design are the same as the system itself, but it is of course possible to include an additional pre-filter on the reference signal.

A common requirement in the control system design is to have integral action, and there are alternative ways to obtain this property. One method is to introduce the integral of the control error as an extra state in the model, while another method is to extend the model with an extra state representing a constant load disturbance. Both cases lead to an extended state space model described by matrices $\bar{A}, \bar{B}$ and $\bar{C}$ that are used in the LQG-design. Further aspects of the integral action will be discussed below.

8.3 Example

In this section two different LQG regulators will be designed. The designs will be carried out for the numerical values of the two-mass model given in Table 8.1. The difference in order of magnitude compared to the identified values in Section 5.3.1 is due to the scaling of the inertias with the gear box ratio and that this is a larger robot.

The aim of the LQG-design is to show two qualitatively different regulators, where one regulator is stable and one is unstable. The regulators will be obtained by

J_m	0.0043	J_a	0.0762
f_m	0.02	f_a	0.005
k	43	d	0.05

Table 8.1 *Parameter values.*

using the same feedback gain L and varying the state estimator gain K. The gain vector L is determined by minimizing the criterion (8.17) using the design variables

$$Q_1 = \text{diag}(100 \;\; 1 \;\; 0 \;\; 0 \;\; 0) \qquad Q_2 = 1$$

The state estimator gains are obtained by computing the Kalman filter gain in the two cases

$$(i) \qquad R_1 = \text{diag}(0 \;\; 0 \;\; 0 \;\; 0 \;\; 1) \qquad R_2 = 10^{-6}$$

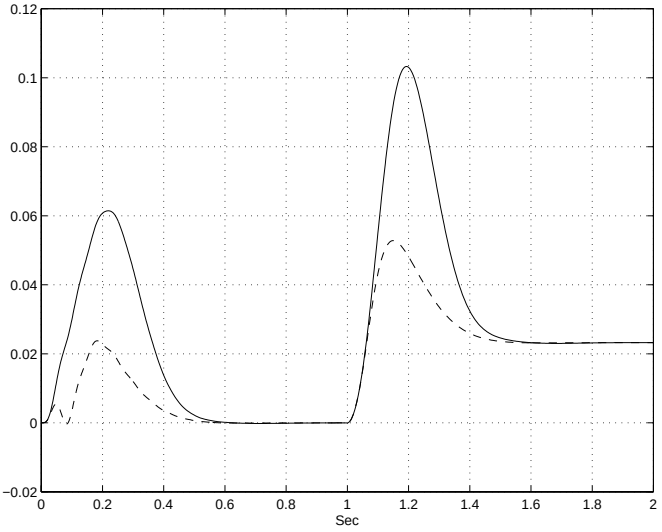
and

$$(ii) \qquad R_1 = \text{diag}(0 \;\; 0 \;\; 0 \;\; 0 \;\; 1) \qquad R_2 = 10^{-8}$$

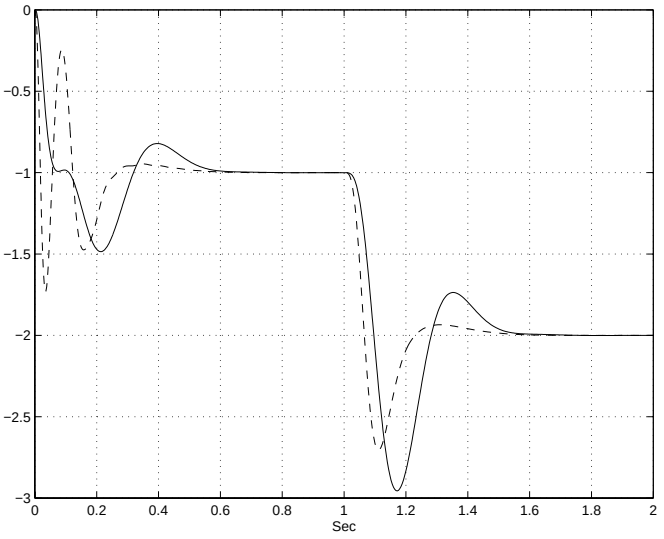
respectively. In order to obtain a regulator with integral action an extra state x_5 is introduced in the model. This state represents a constant load disturbance acting on the second mass. Case (i) gives a stable regulator, while case (ii) gives an unstable one, where $F_y(s)$ has two poles in the right half plane. To illustrate the difference in performance a simulation experiment is carried out where step disturbances in both $w(t)$, acting on the first mass, and $v(t)$, acting on the second mass, are applied. The disturbances have unit amplitude, and they are applied at $t = 0$ and $t = 1$ seconds respectively. The angle of the second mass $\varphi_a(t)$ is shown in Figure 8.3(a), and the figure shows that the unstable regulator gives considerably better rejection of the disturbances. The control signal in the two cases are also shown in Figure 8.3(b).

8.4 Regulator stability

It is well known that conventional linear control design methods may result in regulators whose transfer functions are unstable, i.e., the situation that $F_y(s)$ has poles in the right half plane. One situation when this happens is when the poles and the zeros of the system to be controlled are located on the positive real axis in a particular pattern. In such a situation it is necessary to use an unstable regulator in order to achieve a stable closed loop system (see, e.g., Youla et al., 1974). Another situation, which is the one that will be considered here, is when unstable regulators appear also when open loop stable systems are considered. Examples of this situation can be found in Hagander and Bernhardsson (1990),



(a) Response in $\varphi_a(t)$ to step disturbances in $w(t)$ and $v(t)$.



(b) Control signal in step disturbance simulations.

Figure 8.3 *Step disturbances in $w(t)$ and $v(t)$. Solid: Stable regulator, case (i). Dashed: Unstable regulator, case (ii).*

Hippe and Wurmthaler (1999), Wallenborg and Åström (1988), Gunnarsson and Östring (1999). In the example above the regulator became unstable when the performance requirements were increased, and it will now be indicated that this is a general property.

Let the requirements on the closed loop system from reference signal to φ_a be specified by a complementary sensitivity function $T_a(s)$ that has a certain bandwidth. This implies a desired complementary sensitivity function to φ_m given by

$$T_m(s) = T_a(s)G_a^{-1}(s) \tag{8.26}$$

where G_a is the transfer function from φ_m to φ_a. This implies that, provided that $G_{um}^{-1}(s)$ is stable, the regulator shall be chosen as

$$F_y(s) = \frac{T_m(s)}{1 - T_m(s)} G_{um}^{-1}(s) \tag{8.27}$$

Here $G_{um}(s)$ is the transfer function from input torque to φ_m, i.e.,

$$G_{um}(s) = \frac{G_m(s)}{1 - G_m(s)G_a(s)G_1(s)} \tag{8.28}$$

The regulator $F_y(s)$ is stable if the denominator $1 - T_m(s)$ does not have any zeros in the right half plane. This is ensured if $T_m(i\omega)$ does not encircle the point $+1$ in the complex plane, or, equivalently, if $T_a(i\omega)G_a^{-1}(i\omega)$ does not encircle this point. For the numerical example given by Table 8.1 the Bode diagram of $G_a(s)$ is shown in Figure 8.4. From the figure it is found that the gain of $G_a^{-1}(s)$ increases for ω above the resonance peak of $G_a(s)$. This implies that when the bandwidth of $T_a(s)$ is chosen too large the gain of $T_a(i\omega)G_a^{-1}(i\omega)$ will be greater than one for a large region. Therefore it is a risk that the point $+1$ will be encircled. Figure 8.5 shows an example where $T_a(s)$ is a third order system with a bandwidth of approximately 50 rad/s. The Nyquist curve encircles $+1$ which implies that an unstable regulator is necessary in order to obtain the desired bandwidth.

8.5 Input saturation

An important aspect of the occurrence of unstable regulators is the interaction with nonlinearities in general and input saturation in particular. This is an important aspect since all control systems in reality are subject to input limitations. This topic will be discussed in the situation when the control system is based on feedback from estimated states. The system contains an input nonlinearity which means that the applied control signal is given as

$$u(t) = f(\bar{u}(t)) \tag{8.29}$$

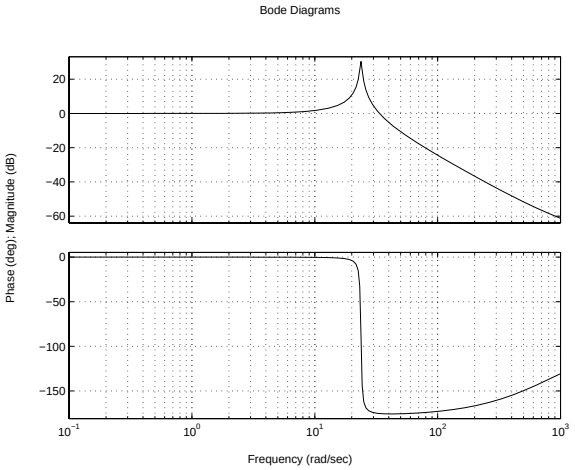


Figure 8.4 *Bode diagram of* $G_a(s)$.

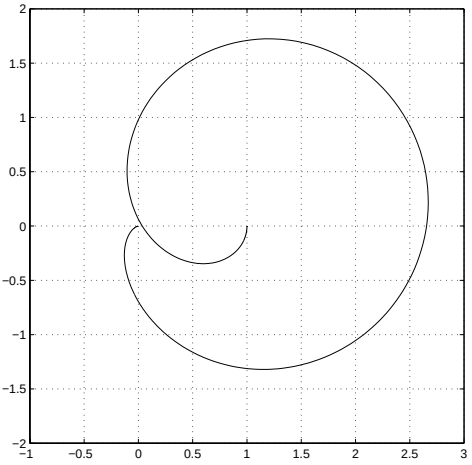


Figure 8.5 *Nyquist diagram of* $T_a(i\omega)G_a^{-1}(i\omega)$.

where $\bar{u}(t)$ denotes the computed control signal. Using the computed control signal $\bar{u}(t)$ in the state estimator implies that the effects of the saturation shall be investigated using the the Nyquist curve of $F_y(s)G(s)$, where $G(s)$ is the transfer function of the system. This can be done by, for example, applying the describing

function method (see, e.g., Cook, 1986). The describing function of a saturation is

$$Y_f(C) = \begin{cases} \frac{2}{\pi}(\arcsin\frac{1}{C} + \frac{1}{C}\sqrt{1-\frac{1}{C^2}}) & C > 1 \\ 1 & C \leq 1 \end{cases} \tag{8.30}$$

which implies that $-1/Y_f(C)$ is the line from -1 towards $-\infty$. The Nyquist curves of the loop gain are given in Figure 8.6, and it is obvious that there will be an intersection between the Nyquist curve and $-1/Y_f(C)$ slightly to the left of -1. This indicates that there is a problem with oscillations, and this is verified in simulations. It is worth noticing that also for the stable case the Nyquist curve intersects the negative real axis.

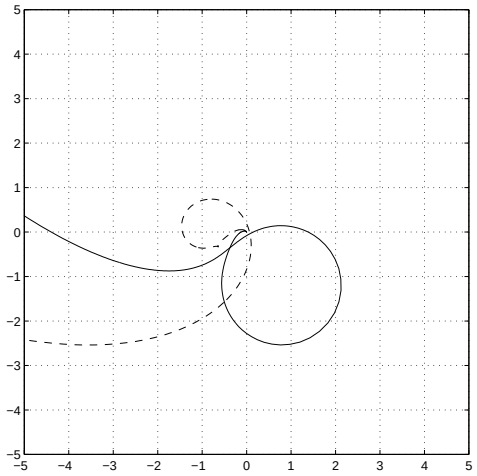


Figure 8.6 *High frequency part of the Nyquist curves $F_y(i\omega)G(i\omega)$. Solid: Stable regulator, case (i). Dashed: Unstable regulator, case (ii).*

There are a large number of publications dealing with the problem of integrator windup. A survey of different approaches is given in Edwards and Postlethwaite (1996). The method that will be studied here, which is described in more detail in Åström and Wittenmark (1984), is to use the applied control signal in the state estimator. The system is given by Equations (8.2) and (8.3) while the input is generated by

$$\bar{u} = -\bar{L}\hat{z} \tag{8.31}$$

where $\hat{z}$ is the estimate of the extended state vector and generated by

$$\dot{\hat{z}} = \bar{A}\hat{z} + \bar{B}u + \bar{K}(y - \bar{C}\hat{z}) \tag{8.32}$$

Using Laplace transforms the computed input is given by

$$\bar{U}(s) = -\bar{L}(sI - \bar{A} + \bar{K}\bar{C})^{-1}[\bar{B} + \bar{K}C(sI - A)^{-1}B]U(s) \tag{8.33}$$

Since the introduced disturbance state is observable but uncontrollable the transfer function of the original and extended systems will be the same. The expression $C(sI - A)^{-1}B$ in the equation above can hence be replaced by $\bar{C}(sI - \bar{A})^{-1}\bar{B}$. It is then straightforward to show that

$$\bar{U}(s) = -G_0(s)U(s) \tag{8.34}$$

where

$$G_0(s) = \bar{L}(sI - \bar{A})^{-1}\bar{B} \tag{8.35}$$

In the derivation of (8.35) it is also assumed that the eigenvalues of $\bar{A} - \bar{K}\bar{C}$ are strictly in the left half plane. The describing function method shall hence be applied using the Nyquist curve of $G_0(s)$ where $\bar{L}$ is the state feedback gain vector obtained using the extended state space model. It should be noted that $G_0(s)$ does not depend on the properties of the state estimator, i.e., whether $F_y(s)$ is stable or not. Figure 8.7 shows the Nyquist curve of G_0 for the gain vector $\bar{L}$ obtained above.

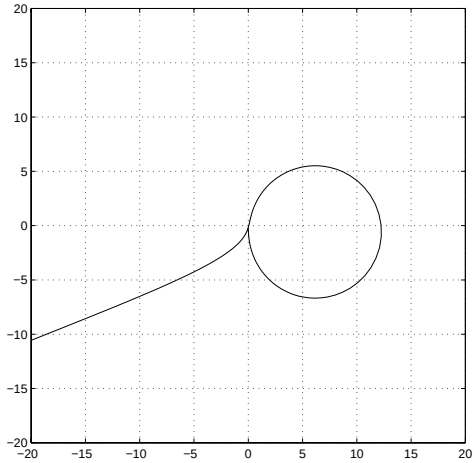


Figure 8.7 *Nyquist curve of* $G_0(i\omega)$.

The intersections between the Nyquist curve and $-1/Y_f(C)$ are no longer present. It should be remembered that the describing function method is an approximate method, and that it cannot be used to prove stability of the nonlinear system. However, it gives an indication of the system behavior. The observation is also

supported by simulation results. The conclusion will therefore be that the use of unstable regulators will require a properly designed anti-windup method (see, e.g., Edwards and Postlethwaite, 1996).

8.6 Robustness

A further interesting aspect of the use of unstable regulators is to investigate how the robustness properties of the control system are affected. This aspect can be studied by looking, in a Nyquist diagram, at the distance between the Nyquist curve of the open loop system and the point -1. Figure 8.6 shows the Nyquist curves of $F_y(i\omega)G(i\omega)$ for the stable and unstable cases respectively. Since the loop gain has poles in the right half plane the Nyquist criterion implies that the Nyquist curve has to encircle the point -1 sufficiently many times. In this particular example $F_y(s)$ has two unstable poles, and hence the Nyquist curve encircles -1 twice.

A more general observation of the effects of using an unstable regulator can be made using Bode's integral theorem, see, for example, Maciejowski (1989). The theorem states that for a control system where $F_y(s)G(s)$ has the poles $p_1, \ldots p_M$ in the right half plane and for high frequencies decays like $1/s^p$ where $p \geq 2$ the sensitivity function

$$S(s) = \frac{1}{1 + F_y(s)G(s)} \tag{8.36}$$

satisfies the relationship

$$\int_0^\infty \log \mid S(i\omega) \mid d\omega = \pi \sum_{i=1}^{M} Re(p_i) \tag{8.37}$$

The connection between this result and the observation in the Nyquist diagram above becomes clear by noting that the distance between the Nyquist curve and -1 is the same as the inverse of the sensitivity function, i.e.,

$$\mid 1 + F_y(i\omega)G(i\omega) \mid = \frac{1}{\mid S(i\omega) \mid} \tag{8.38}$$

When the loop gain has unstable poles, i.e., the right hand side of equation (8.37) is larger than zero, the interval where $\mid S(i\omega) \mid > 1$ will be comparatively larger. Hence the interval where the distance from the Nyquist curve to -1 is less than one will also be larger. Furthermore the maximum value of the sensitivity function will be inversely proportional to the minimum distance. All these properties are illustrated in Figure 8.8, which shows the absolute value of the sensitivity functions in the stable and unstable cases. The frequency range, in the unstable case, where

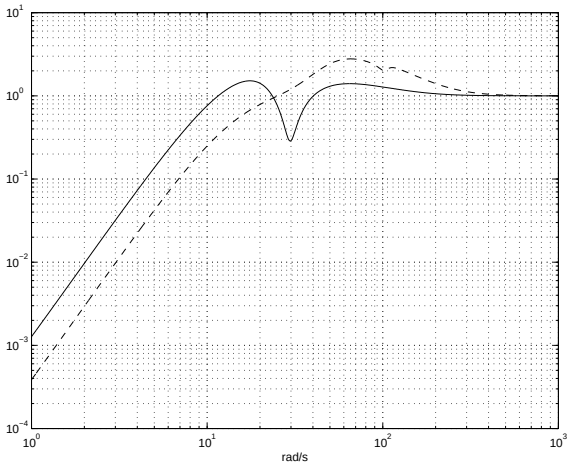


Figure 8.8 *Sensitivity function. Solid: Stable regulator, case* (i)*. Dashed: Unstable regulator, case* (ii)*.*

the absolute value of the sensitivity function is around two corresponds to the frequency range in Figure 8.6 where the distance to -1 is around one half.

The conclusion of this section hence is that an unstable regulator implies a Nyquist curve that is close to -1 for a large frequency interval. This implies that it is necessary to have a very accurate model in order to succeed with the regulator design.

8.7 LQG control using acceleration feedback

In the discussion above only the motor angle $\varphi_m(t)$ has been used in the control system while the requirements concerning response speed and damping are formulated for the arm angle $\varphi_a(t)$. Absolute measurement of the arm angle in industrial practice is rather difficult while the use of accelerometers is more realistic. In this section it shall be investigated if there are any benefits by using the arm angle acceleration $\ddot{\varphi}_a(t)$ in the control system.

Recall the state space representation of the two-mass model

$$\dot{x}(t) = Ax(t) + Bu(t) + B_w w(t) + B_v v(t) \tag{8.39}$$

where the matrices $A,\ B,\ B_w$ and B_v are given in (8.4) and (8.5). Let now $y(t)$

denote a column vector containing the two measured signals, i.e.,

$$y(t) = \begin{pmatrix} \varphi_m(t) \\ \ddot{\varphi}_a(t) \end{pmatrix} \tag{8.40}$$

The measured signals are then related to the states according to

$$y(t) = Cx(t) \tag{8.41}$$

where

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ k/J_a & d/J_a & -k/J_a & -(d+f_a)/J_a \end{pmatrix} \tag{8.42}$$

The LQG method can be applied directly to this model with two measured signals with appropriate choice of covariance matrices in the Kalman filter design. Also here it is of interest to incorporate integral action such that zero steady state error in motor position is obtained when constant disturbances are acting on the system. Therefore the method with an extra state representing load disturbance is applied also in this case.

In the simulations shown below the following design variables were chosen as

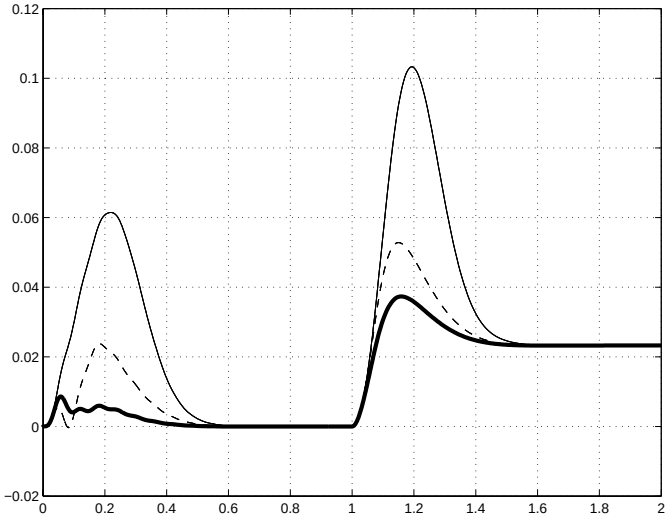
$$Q_1 = \text{diag}(100 \;\; 1 \;\; 0 \;\; 0 \;\; 0) \qquad Q_2 = 1$$

and

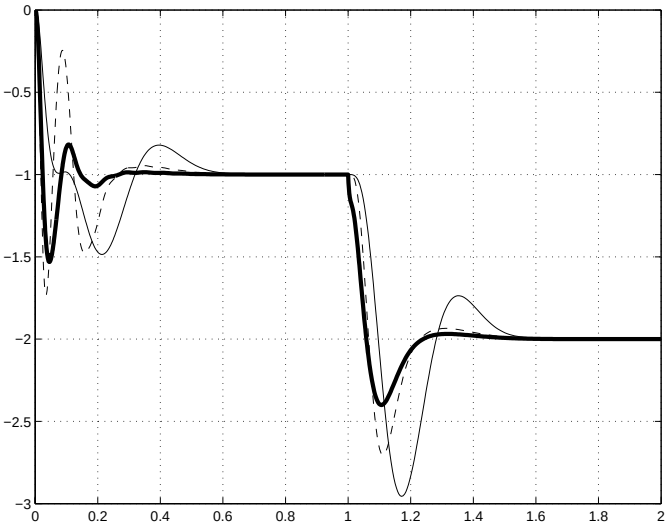
$$R_1 = \text{diag}(0 \;\; 0 \;\; 0 \;\; 0 \;\; 1) \qquad R_2 = \text{diag}(10^{-8} \;\; 10^{-1})$$

respectively. The LQG control was evaluated by applying step signals in $w(t)$ and $v(t)$.

Figure 8.9(a) shows the response in arm and motor angle to a step in $w(t)$ and $v(t)$, i.e., the disturbance acting on the motor side and on the arm side. The figure shows that the peak value of the arm angle has been reduced by approximately 20 % while the settling time is about the same. The peak value of the arm angle has been reduced substantially, approximately by a factor three. Also the settling time has been slightly reduced. In Figure 8.9(b) the input (torque) signal $u(t)$ is also shown in the two cases. Note that the regulator using acceleration feedback is stable. This implies that the sensitivity function can be lower than in the case with the unstable regulator according to (8.37). Figure 8.10 shows the sensitivity function with and without the acceleration feedback.



(a) Response in $\varphi_a(t)$ to a step in $w(t)$ and $v(t)$.



(b) Applied torque $u(t)$ caused by a step in $w(t)$ and $v(t)$.

Figure 8.9 *A step in $w(t)$ and $v(t)$. Bold solid: With acc feedback (stable regulator). Dashed: Without acc feedback (unstable regulator). Solid: Without acc feedback (stable regulator).*

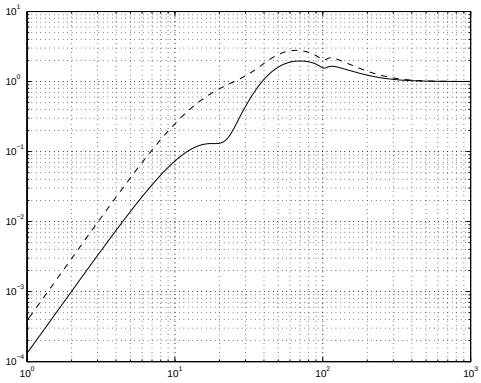


Figure 8.10 *Sensitivity function. Solid: With acc feedback (stable regulator). Dashed: Without acc feedback (unstable regulator)*

8.8 Summary

LQG control of a two-mass approximation of a flexible robot are has been evaluated with respect to the ability to reduce the influence of load disturbances on the second mass. In a first stage only measurements of the angle of the first mass (motor) were used. A main observation was that when the performance requirements were increased too much the resulting regulator became unstable. Different aspects of this phenomenon were investigated, with special attention to robustness and ability to cope with input saturation. In the second stage the angular acceleration of the second mass was included in the LQG regulator. Including this signal it was found that good disturbance rejection properties could be obtained while the stability of the regulator was maintained.

9 Conclusions

In this chapter a summary and a reflection of the results and conclusions in the thesis are given. Some parts that will need further work are also pointed out.

9.1 Summary

This thesis is a small step towards the future vision that was painted in the introduction, where a diagnosis system is monitoring each robot in a big plant, and the robot can diagnose and estimate faults in model parameters by itself.

The focus of the thesis is on the robot application although most of the methods used can be applied to other systems as well. The identification of the physical parameters is a general method that works for most mechanical systems and many other systems too. The same is true for the recursive counterpart and the diagnosis. The only severe limitation is that the faults, that are being detected, isolated, and identified, must be described by parameter changes.

The thesis shows that it is possible to estimate the physical parameters of a flexible arm structure using only measurements from the motor side. It also shows that

the estimation can be made recursive. The flexible models can then be used in the isolation and detection of faults modeled using parameter changes.

In the last part of the thesis, limitations on disturbance rejection using LQG control of a flexible mechanical system is studied. The conclusion is that there is a fundamental limitation on the performance of the controller if it has to be stable. Another conclusion is that an improved disturbance rejection can be acquired if an additional accelerometer sensor is used.

9.2 Further work

The models in the modeling chapter can be extended. A first step can be to include nonlinearities such as the Coloumb friction. If this is modeled and identified it is possible to compensate for the Coloumb friction previous to the identification of the linear parts of model. The next step can be to look at backlash and nonlinear springs. It is also possible to look at static nonlinearities of the inputs and outputs so called Hammerstein and Wiener models (Hagenblad, 1999). Another are that needs further research is how to find good initial values of the physical parameters in the off-line identification. This is especially important when many masses are used in the model.

In the recursive identification the estimation of time varying parameters can be improved by changing the adaption and/or running multiple estimators at the same time according to different hypothesis of, for example, change time. There exist also algorithms which use time varying design parameters to keep the information content approximately constant, that is varying the forgetting. This would also be an interesting subject for further research.

The diagnosis algorithms do not take the correlation between parameters into account. How to use this information can be a topic for further research. In the diagnosis part it would be interesting to use data originating from actual faults introduced in the system. In many cases this means to change some of the physics on-line, which may be difficult. Other faults and other fault models, perhaps described by signal faults, would also be interesting to study. The area of model based fault diagnosis is wide, and there are other methods that can be tested and evaluated. Dealing with under modeling is also an area that is left for further work, see for example Juričić and Žele (2002).

An interesting area for further research is to extend the models to more than one link, especially for the on-line algorithms. This includes areas such as modeling of the coupling between different links and dealing with known time varying inertias.

The work in this thesis is made with one of the smallest ABB robots, ABB IRB 1400. It would be interesting to study robots using other types of mechanical structures. An example would be to study larger robots where the flexibilities are causing more control related trouble, especially when dealing with large loads where the flexibilities are more prominent.

Notation

Abbreviations and acronyms

ARX	Auto-Regressive with eXogenous Input.
BJ	Box-Jenkins model.
BOOT-server	Software for booting computers over the network.
CUSUM	Cumulative sum algorithm.
DOF	Degrees Of Freedom.
DSP	Digital Signal Processor.
ETFE	Empirical Transfer Function Estimate.
FDI	Fault Detection and Isolation.
FFT	Fast Fourier Transform.
FIR	Finite Impulse Response model.
FTP	File Transfer Protocol.
ILC	Iterative Learning Control.
IRB	Industrial RoBot.
ISIS	Information Systems for Industrial Control and Supervision
LQG	Linear Quadratic Gaussian.
MATLAB™	Software for technical computing.

NFS	Network File System.
OE	Output Error model.
RAPID	Programming language for ABB Robots.
RIA	Robot Institute of America.
RPEM	Recursive Prediction Error Methods.
SITB	MATLAB^{TM} System Identification ToolBox, (Ljung, 2000)

Symbols

$\alpha_k(\omega)$	Weighting function.
$\varepsilon(t)$	Prediction error $y(t) - \hat{y}(t, \theta)$.
$e(t)$	Noise variable (usually white).
$\hat{G}(e^{i\omega})$	Estimate of the transfer function.
$G(q, \theta)$	Model of the system parameterized by θ.
$G_{\theta^{(i)}}(q, \theta)$	Derivative of $G(q, \theta)$ w.r.t. $\theta^{(i)}$.
$G_0(q)$	True system.
$H(q, \theta)$	Noise model parameterized by θ.
$H_0(q)$	True noise model.
$h_i(t)$	Adaptive threshold for the ith fault.
λ	Forgetting factor.
λ_0	Variance of the noise.
$\hat{\sigma}_i(t)$	Estimated standard deviation of the ith parameter.
$\psi(t)$	Negative gradient of the prediction error $\varepsilon(t)$.
θ	Parameter vector.
$\theta^{(i)}$	The ith parameter in the parameter vector θ.
$T_i(t)$	Test variable for the ith fault.
$u(t)$	The input to the system.
$U_N(\omega)$	Discrete Fourier transform of $u(t)$.
$V_N(\theta)$	Loss function or criterion function.
$v(t),\ w(t)$	Disturbance variable at time t (usually filtered white noise).
$y(t)$	The output from the system.
$\hat{y}(t, \theta)$	One-step-ahead predictor.
$\bar{y}(t)$	The mean of $y(t)$.
$Y_f(C)$	Describing function.
$Y_N(\omega)$	Discrete Fourier transform of $y(t)$.
Z_N	Data set $y(1), u(1), ..., y(N), u(N)$.
$\Phi_u(\omega)$	Spectrum of $u(t)$.
$\Phi_{yu}(\omega)$	The cross spectrum.

Operators

q^{-1}	Delay operator, $q^{-1}u(t) = u(t-1)$.
Ex	Expectation of the random variable x.
$\bar{E}f(t)$	$\lim_{N\to\infty} \frac{1}{N} \sum_{t=1}^{N} Ef(t)$.

Model symbols

φ_m	motor angle.
φ_g, φ_a, φ_p	arm angles.
f_m	friction coefficient of the motor.
k_g, k_a, k_p	spring constants.
d_g, d_a, d_p	damping coefficients in the springs.
J_m	moment of inertia of the motor.
J_g, J_a, J_p	moments of inertia of the arm.
τ	motor torque.
r	gear box ratio ($\frac{1}{118}$ for axis one of IRB 1400).
k_T	static gain.

Bibliography

ABB Flexible Automation (1997a). *Product manual, BaseWare OS 3.0.* ABB Robotics Products AB, Västerås, Sweden. Article number: 3HAC 0808-1.

ABB Flexible Automation (1997b). *User's Guide, BaseWare OS 3.0.* ABB Robotics Products AB, Västerås, Sweden. Article number: 3HAC 0930-1.

Albu-Schäffer, A. and Hirzinger, G. (2001). Parameter identification and passivity based joint control for a 7DOF torque controlled light weight robot. In *International Conference on Robotics & Autmation*, Seoul, Korea.

Anderson, B. and Moore, J. (1979). *Optimal Filtering.* Prentice Hall, Englewood Cliffs, New Jersey, USA.

Asimov, I. (1950). *I, Robot.* Bantam Books.

Åström, K. J. and Wittenmark, B. (1984). *Computer Controlled Systems: Theory and Design.* Prentice-Hall, Englewood Cliffs, New Jersey.

Berglund, E. and Hovland, G. E. (2000). Automatic elasticity tuning of industrial robot manipuators. In *IEEE Conference on Decision and Control*, Sydney, Australia.

Bøgh, S. (1995). Multiple hypothesis-testing approach to FDI for the industrial actuator benchmark. *Control Engineering Practice*, 3(12):1763–1768.

Bolmsjö, G. S. (1992). *Industriell robotteknik*. Studentlitteratur, Sweden.

Broussard, K. and Trahan, R. (1991). Automatic control system failure detection via parameter identification techniques. In *IEEE Proceedings of Southeastcon*, volume 1, pages 176–180.

Chen, J. and Patton, R. (1999). *Robust model-based fault diagnosis for dynamic systems*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Chiang, L., Russell, E., and Braatz, R. (2001). *Fault Detection and Diagnosis in Industrial Systems*. Springer-Verlag.

Cook, P. (1986). *Nonlinear Dynamical Systems.* Prentice-Hall International (UK) Ltd, London.

Dixon, W. E., Walker, I., Dawson, D. M., and Hartranft, J. P. (2000). Fault detection for robot manipulators with parametric uncertainty: A prediction error based approach. *IEEE Transactions on Robotics and Automation*, 16:689–699.

Dépincé, P. (1998). Parameter identification of flexible robots. *IEEE Proceedings on Robotics and Automation*, pages 1116–1121.

Dymola (2002). Dynasim AB. http://www.dynasim.se.

Edwards, C. and Postlethwaite, I. (1996). Anti-windup and bumpless tranfer schemes. In *UKACC International Conference on CONTROL'96*, pages 394–399.

ElMaraghy, W. H., ElMaraghy, H. A., Zaki, A., and Massoud, A. (1994). A study on the design and control of robot manipulators with flexibilities. In *Fourth IFAC Symposium on Robot Control*, pages 495–501.

Forssell, U. (1999). *Closed-loop Identification: Methods, Theory, and Applications*. PhD thesis, Linköpings universitet. Linköping Studies in Science and Technology. Thesis No 566.

Frank, P. and Ding, X. (1997). Survey of robust residual generation and evaluation methods in observer-based fault detection systems. *Journal of Process Control*, 7(6):403–424.

Friedland, B. (1986). *Control system design. An introduction to state space methods.* McGraw-Hill, New York, USA.

Gautier, M. and Poignet, P. (2001). Extended kalman filtering and weighted least squares dynamic identification of robot. *Control Engineering Practice*, 9:1361–1372.

Gertler, J. (1998). *Fault Detection and Diagnosis in Engineering Systems*. Marcel Dekker, Inc, 270 Modison Avenue, New York, USA.

Grotjahn, M., Daemi, M., and Heimann, B. (2001). Friction and rigid body identification of robot dynamics. *International Journal of Solids and Structures*, 38:1889–1902.

Gunnarsson, S. (1996). Combining tracking and regularization in recursive least squares identification. In *Proceedings of the 35th IEEE Conference on Decision and Control*, pages 2551–2552, Kobe, Japan.

Gunnarsson, S. and Östring, M. (1999). On LQG control of a flexible servo. Technical Report LiTH-ISY-R-2140, Department of Electrical Engineering, Linköpings universitet, S-581 83 Linköping, Sweden.

Gunnarsson, S. and Östring, M. (2001). On regulator stability in control of flexible mechanical systems. In *Proceedings of the 32nd International Symposium on Robotics, ISR*, Seoul, Korea.

Gustafsson, F. (2000). *Adaptive Filtering and Change Detection*. John Wiley & Sons, LTD.

Hagander, P. and Bernhardsson, B. (1990). On the notion of strong stabilization. *IEEE Transactions on Automatic Control*, 35:927–92.

Hagenblad, A. (1999). Aspects of the identification of wiener models. Licentiate thesis LIU-TEK-LIC-1999:51 Linköping Studies in Science and Technology. Licentiate Thesis no. 793, Department of Electrical Engineering, Linköpings universitet, SE-581 83 Linköping, Sweden.

Hippe, P. and Wurmthaler, C. (1999). Systematic closed-loop design in the precense of input saturations. *Automatica*, 35:689–695.

Hovland, G., Berglund, E., and Hanssen, S. (2001). Identification of coupled elastic dynamics using inverse eigenvalue theory. In *Proceedings of the 32nd International Symposium on Robotics, ISR*, pages 1392–1397, Seoul, Korea.

Huarng, K. and Yeh, C. (1992). Continous-time recursive least-squares algorithms. *IEEE Transaction on circuits and systems II: Analog and digital signal processing*, 39(10):741–745.

IETF Secretariat (2001). Network file system (NFS). http://www.nfsv4.org.

Isermann, R. (1984). Process fault detection based on modeling and estimation methods – a survey. *Automatica*, 20(4):387.

Isermann, R. (1997). Supervision, fault-detection and fault-diagnosis methods – an introduction. *Control Engineering Practice*, 5(5):639–652.

Isermann, R. and Ballé, P. (1997). Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Engineering Practice*, 5(5):709–719.

Jirstrand, M. (2000). Mathmodelica – a full system simulation tool. In *Product Models 2000, The 6th Conference on Product Models, Global Product Development*, Linköping, Sweden.

Johansson, R., Robertsson, A., Nilsson, K., and Verhaegen, M. (2000). State-space system identification of robot manipulator dynamics. *Mechatronics*, 10:403–418.

Juričić, D. and Žele, M. (2002). Robust detection of sensor faults by means of a statistical test. *Automatica*, 38:737–742.

Kozlowski, K. (1989). *Modelling and Identification in Robotics*. Springer-Verlag.

Ljung, L. (1999). *System Identification: Theory for the User*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2:nd edition.

Ljung, L. (2000). *System Identification Toolbox – User's Guide*. The MathWorks, Inc., Sherborn, MA. USA.

Ljung, L. and Glad, S. T. (1994). *Modeling of Dynamic Systems*. Prentice Hall, Englewood Cliffs, New Jersey, USA.

Ljung, L. and Gunnarsson, S. (1990). Adaption and tracking in system identification – a survey. *Automatica*, 26(1):7–21.

Ljung, L. and Söderström, T. (1983). *Theory and Practice of Recursive Identification*. The MIT Press.

Maciejowski, J. (1989). *Multivariable Feedback Design*. Addison-Wesley Publishing Company, Reading, MA, USA.

Matlab (2001). *Using Matlab, Version 6.1*. The MathWorks Inc.

Moseler, O. and Iserman, R. (1998). Model-based fault detection for a brushless DC motor using parameter estimation. In *Proceedings of the 24th annual conference of the IEEE Industrial Electronics Society (EICON)*, volume 4, pages 1956–1960.

Nilsson, K. (1996). *Industrial Robot Programming*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

Nissing, D. and Polzer, J. (2000). Parameter identification of a substitution model for a flexible link. In *Proceedings of the American Control Conference*.

Norrlöf, M. (1998). On analysis and implementation of iterative learning contol. Licentiate thesis LIU-TEK-LIC-1998:62 Linköping Studies in Science and Technology. Licentiate Thesis No 727, Department of Electrical Engineering, Linköpings universitet, SE-581 83 Linköping, Sweden.

Norrlöf, M. (1999). Modeling of industrial robots. Technical Report LiTH-ISY-R-2208, Department of Electrical Engineering, Linköpings universitet, SE-581 83 Linköping, Sweden.

Norrlöf, M. (2000). *Iterative Learning Control: Analysis, Design, and Experiments*. PhD thesis, Linköpings universitet, Linköping, Sweden. Linköping Studies in Science and Technology. Dissertations No. 653.

Norrlöf, M., Tjärnström, F., Östring, M., and Aberger, M. (2002). Modeling and identification of a mechanical industrial manipulator. In *Proceedings. of the 15th IFAC Congress*, Barcelona, Spain.

Nyberg, M. (1999). *Model Based Fault Diagnosis: Methods, Theory, and Automotive Engine Applications*. PhD thesis, Linköpings universitet, Linköping, Sweden. Linköping Studies in Science and Technology. Dissertations No. 591.

Östring, M. (1998). Dämpningar av svängningar i robotservon. Reg nr: Lith-isy-ex-1948, Department of Electrical Engineering, Linköpings universitet, SE-581 83 Linköping, Sweden.

Östring, M. and Gunnarsson, S. (1999). LQG control of a flexible servo. In *Second conference on Computer Science and Systems Engineering in Linköping, CCSSE99*, pages 125–132, Linköping, Sweden.

Östring, M. and Gunnarsson, S. (2002). Recursive identification of physical parameters in a flexible robot arm. In *Proceedings of the 4th Asian Control Conference, ASCC*, Singapore, Singapore.

Östring, M., Gunnarsson, S., and Norrlöf, M. (2001a). Closed loop identification of an industrial robot containing flexibilities. Technical Report LiTH-ISY-R-2398, Department of Electrical Engineering, Linköpings universitet, SE-581 83 Linköping, Sweden.

Östring, M., Gunnarsson, S., and Norrlöf, M. (2001b). Closed loop identification of the physical parameters of an industrial robot. In *Proceedings of the 32nd International Symposium on Robotics, ISR*, Seoul, Korea.

Östring, M., Gunnarsson, S., and Norrlöf, M. (2002a). Identification of an industrial robot during one axis movements. *Accepted for publication in Control Engineering Practice*.

Östring, M., Tjärnström, F., and Norrlöf, M. (2002b). Modeling of industrial robot for identification, monitoring, and control. In *Proceedings of the International Symposium on Advanced Control of Industrial Processes*, Kumamoto, Japan.

Otter, M. and Elmqvist, H. (2001). Modelica, language, libraries, tools, workshop and EU-project RealSim. http://www.modelica.org.

Parkum, J. E. (1992). *Recursive Identification of Time-Varying Systems*. PhD thesis, The Technical University of Denmark, Lyngby, Denmark.

Patton, R., Frank, P., and Clark, R. (1989). *Faul Diagnosis in Dynamic Systems, Theory and Application*. Prentice Hall, Englewood Cliffs, New Jersey, USA.

Pham, M. T., Gautier, M., and Poignet, P. (2001). Identification of joint stiffness with bandpass filtering. In *International Conference on Robotics & Autmation*, Seoul, Korea.

Pintelon, R. and Schoukens, J. (2001). *System Identification: A Frequency Domain Approach*. IEEE Press, Piscataway, New Jersey, USA.

Söderström, T. and Stoica, P. (1989). *System Identification*. Prentice Hall, Englewood Cliffs, New Jersey, USA.

Secher, J. (2001). Paradigm shift in robot based automation through Industrial IT. In *Proceedings of the 32nd International Symposium on Robotics, ISR*, pages 705–707, Seoul, Korea.

Spong, M. W. and Vidyasagar, M. (1989). *Robot Dynamics and Control*. John Wiley & Sons.

Swevers, J., Ganseman, C., Tükel, D. B., Schutter, J. D., and Brussel, H. V. (1997). Optimal robot excitation and identification. *IEEE Transactions on Robotics and Automation*, 13(5):730–740.

Tiller, M. (2001). *Introduction to Physical Modeling with Modelica*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Tjärnström, F. (2002). *Variance Expressions and Model Reduction in System Identification*. PhD thesis, Linköpings universitet, Linköping, Sweden. Linköping Studies in Science and Technology. Dissertations No. 730.

Unbehauen, H. and Rao, G. (1987). *Identification of Continuous Systems*. North-Holland, Amsterdam, The Netherlands.

Wallenborg, A. and Åström, K. (1988). Limit cycle oscillations in high performance robot drives. In *CONTROL'88. International Conference on Control, 1988*, pages 444–449.

Wang, Q., Bi, Q., and Zou, B. (1996). Parameter identification of continuous-time mechanical systems without sensing accelerations. *Computers in Industry*, 82:207–217.

Wolfram, S. (1999). *The Mathematica Book*. Cambridge University Press.

Xie, L. L. and Ljung, L. (2002). Estimate physical parameters by black-box modeling. In *Proceedings of the Chinese Control Conference*.

Youla, D., Bongiorno, J., and Jabr, H. (1974). Single-loop feeback stabilization of linear dynamical plants. *Automatica*, 10:159–173.