Data-based LQG control

Master's Thesis by:
W.H.T.M. Aangenent

Report No. DCT 2003.113

Eindhoven, December, 2003

Technische Universiteit Eindhoven
Department of Mechanical Engineering
Section Control Systems Technology

Abstract

In this thesis, an infinite horizon discrete-time data-based linear quadratic gaussian (LQG) controller is developed based on the model-based LQG counterpart.
Standard model-based discrete-time finite horizon optimal control of systems is obtained using a state-space description of the system. A cost function is optimized resulting in two difference Ricatti equations, with which the optimal state feedback and observer gains can be computed. Using closed form solutions to these Ricatti equations and introducing a controller state vector, this algorithm can be rewritten in terms of the Markov parameters of the system.
When these parameters are known, the controller state vector can be estimated by an optimal data-based observer and based on this vector, the optimal control input can be calculated to perform the wanted tasks. These Markov parameters equal the values of the impulse response of a discrete system on the sample times and can be estimated from almost any sequence of input/output data of the system. This is done using ARMarkov representations of the system under consideration. ARMarkov representations can be expressed using input/output data, the Markov parameters and some other coefficients. It is possible this way to estimate the coefficients, including the Markov parameters on-line, based on input/output data.
These parameters can be used to construct a moving horizon data-based LQG controller with which a system can be regulated using only the input and output data of the system, without requiring any model of the system or parametric representation of the controller. It is also possible to regulate the tracking error to zero, so a data-based tracking controller is obtained.
Simulations are performed to compare the model- with the data-based LQG controller. Both results are, in case of time-invariant systems, identical. However, since the data-based controller uses measured input/ouput data, it is able to adapt itself to the actual dynamics. This means that the data-based controller can also be used to control time-varying or non-linear systems, since each horizon, the control action is obtained based on the actual dynamics. The model-based controller can become unstable if the model is not accurate enough. Experiments are done on an experimental setup, namely the RRR robotic arm installed in the DCT lab at the Technische Universiteit Eindhoven, which show promising results.

Contents

Chapter 1

Introduction

In spite of the current state of control theory, a fundamental issue in control remains open: what model is appropriate for control design?
Mathematical models of complex systems are not always reliable and adaptive and robust control methods have not completely overcome the danger of modelling errors. Furthermore, the design of an accurate model from first principles or from system identification, is often a time consuming and not a straightforward task. Therefore, another approach to design controllers for a system is investigated in this thesis. A controller is developed that generates control action to perform regulation and tracking for systems, without the use of any model or preliminary information about that system, but based on observed input-output data only. This way, the step of modelling or system identification is not required anymore. Such a controller belongs to the class of so called data-based controllers.

1.1 Data-based controllers

At this moment, there is no consensus about the meaning of data-based control. The data-based control field encompasses versatile research interests, approaches and case studies. Roughly speaking, in data-based techniques, controllers are designed without explicitly making use of parametric models, but merely based on empirical signals. One typical statement about the difference between model-based and data-based controllers is given by Skelton [20]:

A model-based controller requires a transfer function or a state-space description of the system. A data-based controller requires neither.

The idea is based on the fact that all relevant information about the system to be controlled is included in the signals that can be measured, while model-based control techniques are often based on approximate models which are unable to describe the complete system dynamics. Various approaches to design data-based controllers are possible. One related concept in this field is *unfalsified control* [23], where a set of controllers is proposed and validated on available data from the system. The approach is to look for inconsistencies between the constraints on signals each candidate controller would introduce if it was inserted in the feedback loop, the constraints with the associated performance objectives and constraints implied by the experimental data. When inconsistencies exist among these constraints, the candidate controller is *falsified* and removed from the class. In essence, it is a feedback generalization of the open-loop model validation schemes used in control-oriented identification.
Another data-based approach is called *iterative feedback tuning* [12]. Here, a parametric representation of a restricted complexity controller is used, of which the parameters are iteratively tuned along the gradient direction of a given cost function, using input-output data without prior identification of the plant dynamics.

The *pulse response based control* method [4] also does not require an explicit model of the systems dynamics. The behavior of the system is represented in terms of measured response to pulses in control inputs, rather than in terms of an explicit model. A control profile is obtained that results in prescribed outputs at the end of the control task, according to convolution of the pulse response data.
The *virtual reference feedback tuning* method [6, 7, 15] is a method that directly estimates the parameters of feedback controllers from plant input/output data. A reference model for the closed-loop control system dynamics is introduced together with a virtual reference signal, thanks to which the closed-loop input/output behavior of the overall control system can be shaped (off-line) by using only a single record of input/output open-loop measurements performed on the system.
Other data-based techniques are available and include, for instance, disturbance-based control [21, 22].
All above mentioned concepts however, require a parametric representation of the controller. Skelton [20] defined a controller that generates control action in the form of data points, computed by adequate processing of experimental data. No parametric representation (i.e. a state-space model, a transfer function or an other predefined model with unknown parameters) of the system or the controller is needed, so this is a truly data-based controller. The algorithm results essentially in a discrete-time finite horizon Linear Quadratic Gaussian (LQG) controller, based on the Markov parameters of the controlled system, which can be estimated online from input-output data. This thesis deals with the design and implementation of this controller.

1.2 Problem formulation

In this report, the discrete data-based LQG algorithm presented in [20] is studied and implemented. The goal of the paper from Skelton is to show that the optimal control inputs for a small horizon can be computed using less Markov parameters than are needed to generate a complete model of the system. No implementation issues are discussed, nor is any attempt made to design a complete (in)finite time controller based on the in- and output data. As far as we know, no further research has been done in this specific area in the last years. In order to obtain a working controller for infinite time, the algorithm has to be extended from a finite to an infinite horizon. Furthermore, a way has to be found to estimate the needed Markov parameters on-line, using only input/output data. The objectives can be summarized as follows:

- Study and implement the data-based LQG algorithm from Skelton
- Develop and implement an algorithm which estimates the Markov parameters on-line, based on input/output data of the system.
- Combine these results to obtain an infinite time optimal controller and perform simulations and experiments on an experimental setup: the RRR robotic arm, installed in the Dynamics and Control Laboratory at the Department of Mechanical Engineering at the Technische Universiteit Eindhoven.

1.3 Report overview

The report is organized as follows:

In chapter 2 the model-based LQG control algorithm is revised. The algorithm minimizes a weighted cost function regarding the state and input of the system using a state-space model. This optimization results in a difference Ricatti equation from which the solution is used to compute the optimal state feedback gain (LQ control). In most cases, however, not all states are available for measurement. Therefore, an optimal observer is developed which estimates the states of the system, based on input/output measurements, a state-space description of the system and some

Gaussian statistical properties for system uncertainty and measurement noise. The combination of the optimal controller and observer results in the LQG controller.

Chapter 3 deals with the data-based LQG algorithm. Here, the model-based LQG algorithm is rewritten in terms of the Markov parameters of the system. These Markov parameters can be estimated from input/output data and directly used in the algorithm. A data-based LQG controller is obtained which can be implemented in a recursive manner.

The fourth chapter describes the estimation of the Markov parameters using input/output data of the system. Markov parameters are basically the values of the impulse response of a system. An algorithm is presented with which these parameters can be estimated using almost any sequence of input/output data.

In chapter 5, the implementation of both algorithms is discussed and the infinite horizon data-based controller is synthesized, using the results from chapter three and four. The model- and data-based controllers are compared and some simulations are presented, both on time-invariant and time-varying systems.

Simulations and experiments on the RRR robotic arm are discussed in chapter 6. Three control schemes are analyzed, namely control with the use of feedback linearization, control using model-based feedforward and decentralized control.

The final chapter contains conclusions about the obtained results and provides recommendations for future research.

Chapter 2

Model based LQG control

In this chapter, the optimal control problem is revised. A closed-form solution to the derived Ricatti equations is presented, which is needed for the data-based algorithm and some problems are addressed considering the use of optimal control for tracking purposes. This chapter is included for completeness, the theoretical background needed in the derivation of the data-based LQG algorithm is presented and is taken from [16], [10], [9] and [8].

2.1 Introduction

In optimal control, a weighted cost function, penalizing the state of the system and the control input, is minimized using the system equations. It is an optimization problem which results in a Ricatti equation with which the optimal (time-varying) state feedback gain can be computed, leading to the optimal inputs. Since in general not all states are measured, an observer is used to estimate and reconstruct the states of the system. This observer is also computed using an optimization problem. Since in both optimizations a linear quadratic cost function is used and since the process and measurement noises are assumed to be Gaussian, the optimal control problem combined with the observer is known as Linear Quadratic Gaussian (LQG) control. At first, the general optimal control problem is presented after which the discrete-time Kalman filter and the discrete-time LQG control problem are stated.

2.2 General optimal control

In this section, an optimal control algorithm will be derived from the optimization of a cost function with equality constraints (the system equations). First, the general solution of the optimization problem is presented, therefore the general nonlinear discrete-time dynamical equation

$$x_{k+1} = f_k(x_k, u_k), \tag{2.1}$$

with initial condition x_0 is used. The state x_k is an n dimensional vector while the control input u_k is an m dimensional vector. Since equation (2.1) determines the state at time $k+1$, given the control and state at time k, this equation is the constraint relation. Now a cost function has to be chosen. For this a performance index in the general form

$$J_i = \Phi(N, x_N) + \sum_{k=i}^{N-1} L_k(x_k, u_k), \tag{2.2}$$

is used, where $[i, N]$ is the time interval of interest over which the behavior of the system is analyzed. $\Phi(N, x_N)$ is a function of the final time N and the state at that time and $L_k(x_k, u_k)$ is in general a time-varying function of the state and control input at each intermediate time k in

$[i, N]$. The optimal control problem is to find the control u_k^* on this interval that drives the system (2.1) along a trajectory x_k^* such that the performance index (2.2) is minimized. The stars denote the optimal values. The performance index can be chosen to obtain the desired system response (e.g. minimal time, minimum control effort, minimal energy).
To determine the optimal control sequence $u_i^*, u_{i+1}^*, \dots, u_{N-1}^*$ minimizing J, the Lagrange-multiplier approach is used, which is taken from [16]. The usefulness of this approach lies in the fact that in reality the increments dx and du are not independent. By introducing an undetermined multiplier λ (i.e. the Lagrange multiplier), we obtain an extra degree of freedom and λ can be selected to make dx and du behave as if they were independent increments. The Lagrange multiplier thus replaces the problem of minimizing the performance index with a constraint with the problem of minimizing a Hamiltonian $H(x, y, \lambda)$ without constraints. Since there is a constraint function $f_k(x_k, u_k)$ specified at each time k in the interval of interest, a Lagrange multiplier is required at each time.
To solve this optimization problem, append the constraint (2.1) to the performance index (2.2) to define an augmented performance index J'

$$J' = \Phi(N, x_N) + \sum_{k=i}^{N-1} \left[L_k(x_k, u_k) + \lambda_{k+1}^T (f_k(x_k, u_k) - x_{k+1})\right]. \tag{2.3}$$

With f_k, the multiplier λ_{k+1} is associated in stead of λ_k, this is done since it makes the solution neater. Defining the Hamiltonian function as

$$H_k(x_k, u_k, \lambda_{k+1}) = L_k(x_k, u_k) + \lambda_{k+1}^T f_k(x_k, u_k), \tag{2.4}$$

the cost function becomes

$$J' = \Phi(N, x_N) - \lambda_N^T x_N + H_i(x_i, u_i, \lambda i + 1) + \sum_{k=i+1}^{N-1} \left[H_k(x_k, u_k, \lambda_{k+1}) - \lambda_k^T x_k\right]. \tag{2.5}$$

The increment in J' due to increments in all the variables x_k, u_k and λ_k is examined. Assume the final time N is fixed. According to the Lagrange multiplier theory, at a constraint minimum this increment dJ' should be zero. Therefore

$$\begin{aligned} dJ' = (\Phi_{x_N} - \lambda_N)^T dx_N + (H_{x_i}^i)^T dx_i + (H_{u_i}^i)^T du_i + \sum_{k=i+1}^{N-1} \left[(H_{x_k}^k - \lambda_k)^T dx_k + (H_{u_k}^k)^T du_k\right] \\ + \sum_{k=i+1}^{N} (H_{\lambda_k}^{k-1} - x_k)^T d\lambda_k = 0, \end{aligned} \tag{2.6}$$

where

$$\begin{aligned} \Phi_{x_N} &= \triangleq \frac{\partial \Phi}{\partial x_N}, \\ H_{x_k}^k &\triangleq \frac{\partial H^k}{\partial x_k}, \end{aligned} \tag{2.7}$$

and so on. Necessary conditions for a constrained minimum are thus given by

$$\begin{aligned} x_k &= \frac{\partial H^{k-1}}{\partial \lambda_k}, \quad k = i+1, \dots, N \\ &= f_{k-1}(x_{k-1}, u_{k-1}), \quad k = i+1, \dots, N, \end{aligned} \tag{2.8a}$$

$$\lambda_k = \frac{\partial H^k}{\partial x_k} = \left(\frac{\partial f^k}{\partial x_k}\right)^T \lambda_{k+1} + \frac{\partial L^k}{\partial x_k}, \quad k = i, \dots, N-1, \tag{2.8b}$$

$$0 = \frac{\partial H^k}{\partial u_k} = \left(\frac{\partial f^k}{\partial u_k}\right)^T \lambda_{k+1} + \frac{\partial L^k}{\partial u_k}, \quad k = i, \dots, N-1, \tag{2.8c}$$

which arise from the terms inside the summations and the coefficient of du_i, and

$$\left(\frac{\partial \Phi}{\partial x_N} - \lambda_N\right)^T dx_N = 0, \tag{2.9a}$$

$$\left(\frac{\partial H^i}{\partial x_i}\right)^T dx_i = 0. \tag{2.9b}$$

Equality (2.8a) is just the constraint relation, or the system equation. It is a recursion for the state x_k that develops forward in time. Evidently, (2.8b) is a recursion for λ_k that develops backward in time. The Lagrange multiplier is thus a variable that is determined by its own dynamical equation. It is called the costate of the system and (2.8b) is called the adjoint system. This adjoint system and system (2.8a) are coupled difference equations and define a two-point boundary value problem, since the boundary conditions required for the solution are the initial state x_i and the final costate λ_N. Equation (2.8c) is called the stationarity equation. Equations (2.9) specify the conditions needed to solve the recursions (2.8). The first one holds only at final time $k = N$ while the second only holds at initial time $k = i$.
For each of these equations, there exist two possibilities regarding the initial state. If the initial state is fixed, $dx_i = 0$, so (2.9b) hold regardless of the value of $H^i_{x_i}$. On the other hand, in case the initial state is free, $dx_i \neq 0$ so that (2.9b) demands that

$$\frac{\partial H^i}{\partial x_i} = 0. \tag{2.10}$$

Regarding control applications, the system starts at a known initial state x_i (i.e., the initial state is fixed), so there is no constraint on the value of $H^i_{x_i}$.
There are also two possibilities for the final state x_N. In case of a fixed final state, $dx_N = 0$, since x_N is not free to be varied. On the other hand, there is the free final state situation, then (2.9a) demands that

$$\lambda_N = \frac{\partial \Phi}{\partial x_N}, \tag{2.11}$$

and the terminal condition is value (2.11) of the final costate λ_N instead of the final state x_N. In the case of the fixed final state, the control will be an open-loop solution, while in the case of the free final state the control will be a closed-loop one. Since an open-loop control scheme is not robust in most applications, the final state free approach is used from now on. In the next subsection, the optimal control sequence will be computed for a discrete-time linear quadratic (LQ) regulator.

2.2.1 Discrete-time linear quadratic regulator

The previous section provides the solution to the optimal control problem for nonlinear systems with general performance indices, but explicit solutions for the optimal control are hard to deduce. In this subsection we consider the special case of linear systems with quadratic performance indices since these are normally used for control purposes. Therefore, let the system to be controlled be represented by the linear equations

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k u_k, \\ y_k &= C_k x_k, \end{aligned} \tag{2.12}$$

where x_k is an n dimensional state vector, the control input vector u_k is an m dimensional vector and let the performance index be the quadratic function

$$J_i = \frac{1}{2} x_N^T Q_N x_N + \frac{1}{2} \sum_{k=i}^{N-1} \left(x_k^T Q_k x_k + u_k^T R_k u_k\right), \tag{2.13}$$

defined over the interval of interest $[i, N]$. The initial state of the system is given by x_i. We assume that Q_k and Q_N are symmetric positive semidefinite matrices and that R_k is a symmetric positive definite matrix for all k. The optimization is carried out using the Lagrange-multiplier approach, explained in the previous section. In order to solve the equations (2.8) we need the Hamiltonian function

$$H^k = \frac{1}{2}\left(x_k^T Q_k x_k + u_k^T R_k u_k\right) + \lambda_{k+1}^T \left(A_k x_k + B_k u_k\right), \tag{2.14}$$

which gives for the state, costate and stationarity equations respectively

$$x_{k+1} = \frac{\partial H^k}{\partial \lambda_{k+1}} = A_k x_k + B_k u_k, \tag{2.15}$$

$$\lambda_k = \frac{\partial H^k}{\partial x_k} = Q_k x_k + A_k^T \lambda_{k+1}, \tag{2.16}$$

$$0 = \frac{\partial H^k}{\partial u_k} = R_k u_k + B_k^T \lambda_{k+1}. \tag{2.17}$$

According to (2.17)

$$u_k = -R_k^{-1} B_k^T \lambda_{k+1}, \tag{2.18}$$

and using this equation to eliminate u_k in (2.15) gives

$$x_{k+1} = A_k x_k - B_k R_k^{-1} B_k^T \lambda_{k+1}. \tag{2.19}$$

So now the state and costate equations, with the input u_k eliminated, are (2.19) and (2.16) while the control input is given by (2.18).
Since a closed-loop control scheme is desired, the initial condition is given as x_i and the final state x_N is free so that (2.11) holds and since $\Phi = \frac{1}{2} x_N^T Q_N x_N$ the following holds

$$\lambda_N = Q_N x_N. \tag{2.20}$$

This relation between the final costate and the state is the terminal condition. Assume that a linear relation like (2.20) holds for all times $k \leq N$

$$\lambda_k = X_k x_k, \tag{2.21}$$

for some intermediate sequence of $n \times n$ matrices X_k. This is a valid assumption if we can find a consistent formula for these postulated X_k. Now use (2.21) in (2.19) to obtain

$$x_{k+1} = A_k x_k - B_k R_k^{-1} B_k^T X_{k+1} x_{k+1}. \tag{2.22}$$

Solving for x_{k+1} yields

$$x_{k+1} = \left(I + B_k R_k^{-1} B_k^T X_{k+1}\right)^{-1} A_k x_k, \tag{2.23}$$

which is a forward recursion for the state. Now substitute (2.21) in (2.16) to get

$$X_k x_k = Q_k x_k + A_k^T X_{k+1} x_{k+1}, \tag{2.24}$$

and now use 2.23 to see that

$$X_k x_k = Q_k x_k + A_k^T X_{k+1} \left(I + B_k R_k^{-1} B_k^T X_{k+1}\right)^{-1} A_k x_k. \tag{2.25}$$

Since x_k is generally not zero and this equation holds for all state sequences given any x_i, we obtain

$$X_k = A_k^T X_{k+1} \left(I + B_k R_k^{-1} B_k^T X_{k+1}\right)^{-1} A_k + Q_k, \tag{2.26}$$

or, using the matrix inversion lemma (see appendix A.2)

$$X_k = A_k^T \left(X_{k+1} - X_{k+1} B_k \left(B_k^T X_{k+1} B_k + R_k \right)^{-1} B_k^T X_{k+1} \right) A_k + Q_k. \tag{2.27}$$

This is a backward recursion for the postulated X_k which completely specifies it in terms of X_{k+1}, the known system and the weighting matrices together with the boundary condition (the final state weighting matrix Q_N). Therefore, equation (2.21) holds. Equation (2.27) is a difference Riccati equation which evolves backward in time.
The optimal control sequence is still needed, therefore write (2.18)

$$\begin{aligned} u_k &= -R_k^{-1} B_k^T \lambda_{k+1} = -R_{k+1} B_k^T X_{k+1} x_{k+1} \\ &= -R_{k+1} B_k^T X_{k+1} \left(A_k x_k + B_k u_k \right), \end{aligned} \tag{2.28}$$

or,

$$\left(I + R_k^{-1} B_k^T X_{k+1} B_k \right) u_k = -R_k^{-1} B_k^T X_{k+1} A_k x_k. \tag{2.29}$$

Premultiply by R_k and solve for the control input to obtain

$$u_k = -\left(R_k + B_k^T X_{k+1} B_k \right)^{-1} B_k^T X_{k+1} A_k x_k. \tag{2.30}$$

Now the optimal control problem is solved. The optimal input is given by (2.30) where X_{k+1} is the solution of the difference Riccati equation (2.27) with terminal condition Q_N. However, in most control applications, the complete state x_k, which is necessary to compute the optimal input, cannot be measured. Therefore an estimator should be designed which will give optimal estimates for this state. In the next subsection such an estimator is presented.

2.2.2 Discrete optimal state estimation

In most cases, state feedback cannot be applied by just measuring all states since this requires many sensors or is even impossible. Therefore, the states that are not measured need to be reconstructed (or estimated) in some way. This will be done by the use of a recursive filter since then there is no need to store past measurements. This method is taken from [10]. Consider the discrete system (2.31):

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k u_k + D_k w_k, \\ y_k &= C_k x_k + v_k, \end{aligned} \tag{2.31}$$

where x_k denotes the states at time t_k, y_k is the set of ℓ measurements at this time, arranged in vector form:

$$y_k = \begin{bmatrix} y_{1k} \\ y_{2k} \\ \vdots \\ y_{\ell k} \end{bmatrix}, \tag{2.32}$$

w_k is a zero mean, white sequence of covariance W_k and v_k is a vector of random noise quantities with zero mean and covariance V_k.
Now an algorithm is necessary which will, given the prior estimate $\hat{x}_k(-)$ at time t_k, compute an updated estimate $\hat{x}_k(+)$ based on use of the measurement y_k. The indices $(-)$ and $(+)$, are used to denote the times immediately before and immediately after a discrete measurement. To avoid a growing memory filter this estimator is sought in the linear, recursive form

$$\hat{x}_k(+) = L'_k \hat{x}_k(-) + L_k y_k, \tag{2.33}$$

where L'_k and L_k are, as yet unspecified, time-varying weighting matrices. In order to get optimal estimations, this linear estimator has to be optimized using these weighting matrices. An equation for the estimation error directly after the measurement is taken can be obtained from (2.33) through substitution of the measurement equation from (2.31) and the relations

$$\begin{aligned}\hat{x}_k(+) &= x_k + \tilde{x}_k(+),\\ \hat{x}_k(-) &= x_k + \tilde{x}_k(-),\end{aligned} \tag{2.34}$$

where the tilde denotes the estimation error. This results in

$$\tilde{x}_k(+) = \left[L'_k + L_k C_k - I\right] x_k + L'_k \tilde{x}_k(-) + L_k v_k. \tag{2.35}$$

If $E\left[\tilde{x}_k(-)\right] = 0$, the estimator will be unbiased ($E\left[\tilde{x}_k(+)\right] = 0$) if the term in the brackets is zero since by definition $E\left[v_k\right] = 0$. In that case, the following expression should hold

$$L'_k = I - L_k C_k, \tag{2.36}$$

and the estimator takes the form

$$\hat{x}_k(+) = (I - L_k C_k)\hat{x}_k(-) + L_k y_k = \hat{x}_k(-) + L_k\left[y_k - C_k \hat{x}_k(-)\right]. \tag{2.37}$$

If equations (2.31), (2.34) and (2.37) are combined, the corresponding estimation error is obtained

$$\tilde{x}_k(+) = (I - L_k C_k)\tilde{x}_k(-) + L_k v_k. \tag{2.38}$$

Define the error covariance matrix as

$$Y_k = E\left[\tilde{x}_k \tilde{x}_k^T\right]. \tag{2.39}$$

An expression for the change in this matrix when a measurement is employed is (use (2.38))

$$\begin{aligned}Y_k(+) = E\{(I - L_k C_k)\tilde{x}_k(-)&\left[\tilde{x}_k(-)^T (I - L_k C_k)^T + v_k^T L_k^T\right]\\ &+ L_k v_k\left[\tilde{x}_k(-)^T (I - L_k C_k)^T + v_k^T L_k^T\right]\},\end{aligned} \tag{2.40}$$

the following relations hold by definition as well

$$E\left[\tilde{x}_k(-)\tilde{x}_k(-)^T\right] = Y_k(-), \tag{2.41}$$

$$E\left[v_k v_k^T\right] = V_k, \tag{2.42}$$

and since the measurement errors are uncorrelated

$$E\left[\tilde{x}_k(-) v_k^T\right] = E\left[v_k \tilde{x}_k(-)^T\right] = 0. \tag{2.43}$$

Now equation (2.40) simplifies to

$$Y_k(+) = (I - L_k C_k) Y_k(-) (I - L_k C_k)^T + L_k V_k L_k^T. \tag{2.44}$$

It is time to define a criterion to compute the optimum value for L_k. This criterion is the minimization of a weighted scalar sum of the diagonal elements of the error covariance matrix $Y_k(+)$ since this will lead to a minimum norm of the estimation error vector. The cost function then is

$$J_k = E\left[\tilde{x}_k(+)^T O \tilde{x}_k(+)\right], \tag{2.45}$$

where O is any positive definite matrix, but since the optimal estimate does not depend on this weighting matrix, a good choice is $O = I$ yielding

$$J_k = trace\left[Y_k(+)\right]. \tag{2.46}$$

To compute the minimum, it is necessary to take the partial derivative of J_k with respect to L_k and equate it to zero. To do this, we will make use of the following relation for the partial derivative of the trace of the product of two matrices A and B with B symmetric

$$\frac{\partial}{\partial A}\left[trace(ABA^T)\right] = 2AB. \tag{2.47}$$

From equation (2.44) this results in

$$\frac{\partial[trace(Y_k(+))]}{\partial L_k} = -2(I - L_kC_k)Y_k(-)C_k^T + 2L_kV_k = 0, \tag{2.48}$$

which gives for L_k

$$L_k = Y_k(-)C_k^T\left[C_kY_k(-)C_k^T + V_k\right]^{-1}. \tag{2.49}$$

Now equation (2.49) can be substituted in (2.44) yielding

$$\begin{aligned} Y_k(+) &= Y_k(-) - Y_k(-)C_k^T\left[C_kY_k(-)C_k^T + V_k\right]^{-1}C_kY_k(-) \\ &= \left[I - L_kC_k\right]Y_k(-), \end{aligned} \tag{2.50}$$

which is the optimized value of the updated estimation error covariance matrix. Thus far only the behavior of the state estimates and the covariance matrix across measurements is described. To form the estimate (i.e., the predictable portion) $\hat{x}_k(-)$ given $\hat{x}_{k-1}(+)$ the state transition and input matrices from (2.31) are used, resulting in

$$\hat{x}_k(-) = A_{k-1}\hat{x}_{k-1}(+) + B_{k-1}u_{k-1}. \tag{2.51}$$

The error in the estimate at t_k is derived by subtracting the state equation in (2.31) (at time $t_k(-)$) from (2.51) yielding

$$\tilde{x}_k(-) = A_{k-1}\tilde{x}_{k-1}(+) - D_{k-1}w_{k-1}. \tag{2.52}$$

This equation can also be used to develop a relationship for projecting the error covariance matrix Y from time $t_{k-1}(+)$ to $t_k(-)$ as follows. From equation (2.52)

$$\begin{aligned} \tilde{x}_k(-)\tilde{x}_k(-)^T &= (A_{k-1}\tilde{x}_{k-1}(+) - D_{k-1}w_{k-1})(A_{k-1}\tilde{x}_{k-1}(+) - D_{k-1}w_{k-1})^T \\ &= A_{k-1}\tilde{x}_{k-1}(+)\tilde{x}_{k-1}(+)^T - A_{k-1}\tilde{x}_{k-1}(+)w_{k-1}^TD_{k-1}^T - D_{k-1}w_{k-1}\tilde{x}_{k-1}(+)^TA_{k-1}^T \\ &\quad + D_{k-1}w_{k-1}w_{k-1}^TD_{k-1}^T. \end{aligned} \tag{2.53}$$

Taking the expected value of the terms of equation (2.53) and using the fact that the estimation error at t_k and the noise D_kw_k are uncorrelated, namely

$$E\left[\tilde{x}_k(D_kw_k)^T\right] = 0, \tag{2.54}$$

the equation for projecting the error covariance matrix is found to be (use (2.41))

$$Y_k(-) = A_{k-1}Y_{k-1}(+)A_{k-1}^T + D_{k-1}W_{k-1}D_{k-1}^T. \tag{2.55}$$

The covariance update (2.55) can be transformed to a difference Riccati equation as follows. First use formula (2.50) in (2.55) at time t_{k+1}, then substitute (2.49) yielding

$$\begin{aligned} Y_{k+1}(-) &= A_kY_k(+)A_k^T + D_kW_kD_k^T \\ &= A_k(\left[I - L_kC_k\right]Y_k(-))A_k^T + D_kW_kD_k^T \\ &= D_kW_kD_k^T + A_kY_k(-)A_k^T - A_kY_k(-)C_k^T\left[V_k + C_kY_k(-)C_k^T\right]^{-1}C_kY_k(-)A_k^T, \\ Y_0(-) &= D_0W_0D_0^T. \end{aligned} \tag{2.56}$$

The optimal estimation proces is now completed. To summarize the results, the subscripts for the matrices and the remarks (+) and (−) are left out for notational simplification. Use equations (2.51) and (2.31) in (2.37). Then the optimal state estimation, $\hat{x}_k$, can be obtained from

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L_k(y_k - C\hat{x}_k), \tag{2.57}$$

where the estimator gain L_k is given by

$$L_k = AY_kC^T\left(V + CY_kC^T\right)^{-1}, \tag{2.58}$$

where Y_k is the solution of the difference Riccati equation

$$Y_{k+1} = DWD^T + AY_kA^T - AY_kC^T\left(V + CY_kC^T\right)^{-1}CY_kA^T, \quad Y_0 = DW_0D^T. \tag{2.59}$$

2.3 Discrete LQG control

In the past two subsections, the equations for the optimal control input as a feedback of the states of the system and an optimal estimator which can estimate these states are derived. Here, these results are combined to obtain the discrete linear quadratic gaussian (LQG) regulator. A specific cost function needs to be stated for the control problem. The system to be controlled is a linear time invariant discrete-time system given by

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Dw_k, \\ y_k &= Cx_k + v_k, \end{aligned} \tag{2.60}$$

where x_k is the n dimensional state vector, u_k is the m dimensional control input vector, y_k is the ℓ dimensional output vector of interest, w_k is the n_w dimensional disturbance vector to the system and v_k is the ℓ dimensional disturbance vector corrupting the output whose covariance matrices are given by $W > 0$ and $V > 0$ respectively. The model based LQG problem is the problem of finding the functional

$$u_k = f(u_0, u_1, \ldots, u_{k-1}, y_0, y_1, \ldots, y_{k-1}), \tag{2.61}$$

such that the quadratic cost

$$J = \mathbf{E}\left\{y_N^TQy_N + \sum_{k=0}^{N-1}\left(y_k^TQy_k + u_k^TRu_k\right)\right\}, \tag{2.62}$$

is minimized subject to system (3.1), where Q and R are positive definite symmetric weighting matrices. This cost function is based on the output of the system instead of the states since this cost function will be used in the data-based LQG algorithm. If the performance index (2.62) is rewritten as

$$J = \mathbf{E}\left\{x_N^TC^TQCx_N + \sum_{k=0}^{N-1}\left(x_k^TC^TQCx_k + u_k^TRu_k\right)\right\}, \tag{2.63}$$

and if the results from subsections 2.2.1 and 2.2.2 are used, the problem can be solved. The optimal input is given by (2.30)

$$u_k = -K_k\hat{x}_k, \quad k = 0, 1, 2, \ldots, N-1, \tag{2.64}$$

where

$$K_k = \left(R + B^TX_{k+1}B\right)^{-1}B^TX_{k+1}A, \tag{2.65}$$

where X_{k+1} is the solution of the difference Riccati equation

$$\begin{aligned} X_k &= C^TQC + A^TX_{k+1}A - A^TX_{k+1}B\left(R + B^TX_{k+1}B\right)^{-1}B^TX_{k+1}A, \\ X_N &= C^TQC. \end{aligned} \tag{2.66}$$

The optimal state estimation, $\hat{x}_k$ can be obtained from (2.57)

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L_k(y_k - C\hat{x}_k), \tag{2.67}$$

where the estimator gain L_k is given by

$$L_k = AY_kC^T\left(V + CY_kC^T\right)^{-1}, \tag{2.68}$$

where Y_k is the solution of the difference Riccati equation

$$\begin{aligned} Y_{k+1} &= DWD^T + AY_kA^T - AY_kC^T\left(V + CY_kC^T\right)^{-1}CY_kA^T, \\ Y_0 &= DWD^T. \end{aligned} \tag{2.69}$$

Equations (2.64) - (2.69) state the complete solution to the LQG regulator problem for system (3.1) with associated performance index (2.63). In the next subsections, the closed-form solutions for the difference Riccati equations (2.66) and (2.69) will be given.

2.3.1 Closed form solution to the Riccati equation for the LQ control problem

In this subsection a closed-form, i.e., batch-form, solution to the Riccati equation for the LQG problem will be presented. Again, consider the linear time invariant discrete-time system 2.12

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k, \end{aligned}$$

and let the cost-function to be minimized be (2.62)

$$J = y_N^TQy_N + \sum_{k=0}^{N-1}\left(y_k^TQy_k + u_k^TRu_k\right) = \sum_{k=0}^{N}\left(y_k^TQy_k + u_k^TRu_k\right), \tag{2.70}$$

where Q and R are positive definite symmetric weighting matrices. The closed-form solution can be computed using the algorithm presented in [9]. The output $y_k, 0 \le k \le N$ of the system is related to the input u_k by the following relation

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} x_0 + \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ CB & 0 & \ddots & & \vdots \\ CAB & CB & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2} & \cdots & CB & 0 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}. \tag{2.71}$$

This equation is written as

$$\mathbf{y}_0 = \mathbf{C}_0x_0 + \mathbf{S}_0\mathbf{u}_0, \tag{2.72}$$

where

$$\mathbf{y}_0 = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, \mathbf{u}_0 = \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}, \mathbf{C}_0 = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix}, \mathbf{S}_0 = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ CB & 0 & \ddots & & \vdots \\ CAB & CB & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2} & \cdots & CB & 0 \end{bmatrix}.$$

Using these relations, minimizing the cost function (2.70) becomes minimizing

$$\begin{aligned} J &= \mathbf{y}_0^T\mathbf{Q}_0\mathbf{y}_0 + \mathbf{u}_0^T\mathbf{R}_0\mathbf{u}_0 \\ &= x_0^T\mathbf{C}_0^T\mathbf{Q}_0\mathbf{C}_0x_0 + x_0^T\mathbf{C}_0^T\mathbf{Q}_0\mathbf{S}_0\mathbf{u}_0 + \mathbf{u}_0^T\mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0x_0 + \mathbf{u}_0^T\mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0\mathbf{u}_0 + \mathbf{u}_0^T\mathbf{R}_0\mathbf{u}_0 \\ &= \begin{bmatrix} x_0 \\ \mathbf{u}_0 \end{bmatrix}^T \begin{bmatrix} \mathbf{C}_0^T\mathbf{Q}_0\mathbf{C}_0 & \mathbf{C}_0^T\mathbf{Q}_0\mathbf{S}_0 \\ \mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0 & \mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0 \end{bmatrix} \begin{bmatrix} x_0 \\ \mathbf{u}_0 \end{bmatrix}, \end{aligned} \tag{2.73}$$

where

$$\mathbf{R}_0 = \operatorname{diag}(R, R, \ldots, R), \quad \mathbf{Q}_0 = \operatorname{diag}(Q, Q, \ldots, Q).$$

The minimizing solution of this problem exists the system is stabilizable and if $\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0 \geq 0$ and $\ker(\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0) \subseteq \ker(\mathbf{C}_0^T\mathbf{Q}_0\mathbf{S}_0)$, see [9]. If this is the case, the matrix can be decomposed in its appropriate Schur form (appendix A.1)

$$\begin{aligned} J = &\begin{bmatrix} x_0 \\ \mathbf{u}_0 \end{bmatrix}^T \begin{bmatrix} I & \mathbf{C}_0^T\mathbf{Q}_0\mathbf{S}_0(\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{X}_0 & 0 \\ 0 & \mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0 \end{bmatrix} \begin{bmatrix} I & 0 \\ (\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)^{-1}\mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0 & I \end{bmatrix} \begin{bmatrix} x_0 \\ \mathbf{u}_0 \end{bmatrix} \\ &= x_0^T\mathbf{X}_0x_0 + x_0^T\mathbf{C}_0^T\mathbf{Q}_0\mathbf{S}_0(\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)^{-1}\mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0x_0 + \mathbf{u}_0^T\mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0x_0 \\ &\quad + x_0^T\mathbf{C}_0^T\mathbf{Q}_0\mathbf{S}_0\mathbf{u}_0 + \mathbf{u}_0^T(\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)\mathbf{u}_0, \end{aligned} \tag{2.74}$$

where

$$\mathbf{X}_0 = \mathbf{C}_0^T\mathbf{Q}_0\mathbf{C}_0 - \mathbf{C}_0^T\mathbf{Q}_0\mathbf{S}_0(\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)^{-1}\mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0.$$

To find the minimum, take the derivative of J with respect to $\mathbf{u}_0$

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{u}_0} &= \mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0x_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0x_0 + (\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)\mathbf{u}_0 + (\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)^T\mathbf{u}_0 \\ &= 2\mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0x_0 + 2(\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)\mathbf{u}_0, \end{aligned} \tag{2.75}$$

and equate it to zero

$$\begin{aligned} &\frac{\partial J}{\partial \mathbf{u}_0} = \emptyset \Rightarrow \\ &\quad \mathbf{u}_0 = -(\mathbf{R}_0 + \mathbf{S}_0^T\mathbf{Q}_0\mathbf{S}_0)^{-1}\mathbf{S}_0^T\mathbf{Q}_0\mathbf{C}_0x_0. \end{aligned} \tag{2.76}$$

This is the closed-form solution of the LQ optimal control problem. The closed-form solution of the Riccati equation is obtained as follows: instead of looking at horizon N, note that for a given $x_k, \quad 0 \leq k \leq N-1$,

$$\mathbf{y}_k = \mathbf{C}_kx_k + \mathbf{S}_k\mathbf{u}_k, \tag{2.77}$$

where

$$\mathbf{C}_k = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{N-k} \end{bmatrix} = \begin{bmatrix} C \\ \mathbf{C}_{k+1}A \end{bmatrix},$$

$$\mathbf{S}_k = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ CB & 0 & \ddots & & \vdots \\ CAB & CB & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ CA^{N-1-k}B & CA^{N-2-k} & \cdots & CB & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \mathbf{C}_{k+1}B & \mathbf{S}_{k+1} \end{bmatrix}, \tag{2.78}$$

so X_k becomes

$$X_k = \mathbf{C}_k^T\mathbf{Q}_k\mathbf{C}_k - \mathbf{C}_k^T\mathbf{Q}_k\mathbf{S}_k(\mathbf{R}_k + \mathbf{S}_k^T\mathbf{Q}_k\mathbf{S}_k)^{-1}\mathbf{S}_k^T\mathbf{Q}_k\mathbf{C}_k,$$

which can be written, using the matrix inversion lemma (see appendix A.2), as

$$X_k = \mathbf{C}_k^T(\mathbf{Q}_k^{-1} + \mathbf{S}_k\mathbf{R}_k^{-1}\mathbf{S}_k^T)^{-1}\mathbf{C}_k. \tag{2.79}$$

This is the closed form solution of the Riccati equation. It can be easily verified that this solution indeed satisfies the recursive Riccati equation for this LQ problem (2.66)

$$\begin{aligned} X_k &= C^TQC + A^TX_{k+1}A - A^TX_{k+1}B(R + B^TX_{k+1}B)^{-1}B^TX_{k+1}A, \\ X_N &= C^TQC. \end{aligned}$$

First, decompose $\left[\mathbf{R}_k + \mathbf{S}_k^T\mathbf{Q}_k\mathbf{S}_k\right]^{-1}$ in

$$\begin{aligned} \left[\mathbf{R}_k + \mathbf{S}_k^T\mathbf{Q}_k\mathbf{S}_k\right]^{-1} &= \left[\begin{bmatrix} R & 0 \\ 0 & \mathbf{R}_{k+1} \end{bmatrix} + \begin{bmatrix} 0 & B^T\mathbf{C}_{k+1}^T \\ 0 & \mathbf{S}_{k+1}^T \end{bmatrix}\begin{bmatrix} Q & 0 \\ 0 & \mathbf{Q}_{k+1} \end{bmatrix}\begin{bmatrix} 0 & 0 \\ \mathbf{C}_{k+1}B & \mathbf{S}_{k+1} \end{bmatrix}\right]^{-1} \\ &= \begin{bmatrix} R + B^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}B & B^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1} \\ \mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}B & \mathbf{R}_{k+1} + \mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1} \end{bmatrix}, \end{aligned} \tag{2.80}$$

and using the inverse of the Schur decomposition (see Appendix A.1) we obtain

$$\begin{aligned} &\left[\mathbf{R}_k + \mathbf{S}_k^T\mathbf{Q}_k\mathbf{S}_k\right]^{-1} = \\ &\begin{bmatrix} I & 0 \\ -(\mathbf{R}_{k+1} + \mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1})^{-1}\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}B & I \end{bmatrix} \times \begin{bmatrix} (R + B^TX_{k+1}B)^{-1} & 0 \\ 0 & (\mathbf{R}_{k+1} + \mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1})^{-1} \end{bmatrix} \\ &\times \begin{bmatrix} I & -B^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}(\mathbf{R}_{k+1} + \mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1})^{-1} \\ 0 & I \end{bmatrix}, \end{aligned} \tag{2.81}$$

where

$$X_{k+1} = \mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1} - \mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}(\mathbf{R}_{k+1} + \mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1})^{-1}\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}.$$

Combining the above identities with

$$\mathbf{S}_k^T\mathbf{Q}_k\mathbf{C}_k = \begin{bmatrix} B^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}A \\ \mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}A \end{bmatrix}, \quad \mathbf{C}_k^T\mathbf{Q}_k\mathbf{C}_k = C^TQC + A^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}A, \tag{2.82}$$

leads to the following expression for X_k

$$\begin{aligned}
X_k &= C^TQC + A^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}A \\
&\quad - \Big(A^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}B - A^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}\left[\mathbf{R}_{k+1}+\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}\right]^{-1}\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}B\Big) \\
&\quad \times \left[R + B^TX_{k+1}B\right]^{-1} \\
&\quad \times \Big(B^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}A - B^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}\left[\mathbf{R}_{k+1}+\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}\right]^{-1}\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}A\Big] \\
&\quad - A^T\mathbf{C}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}\left[\mathbf{R}_{k+1}+\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}\right]^{-1}\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}A \\
&= C^TQC + A^TX_{k+1}A - A^TX_{k+1}B\left(R+B^TX_{k+1}B\right)^{-1}B^TX_{k+1}A.
\end{aligned} \tag{2.83}$$

The boundary condition can be found by considering the case when $k = N$. The solution then is

$$X_N = C^TQC, \tag{2.84}$$

which, together with equation (2.83), is exactly the recursive solution to the Riccati equation (compare with equation (2.66)).
The minimizing control at time k, u_k, is the first element from the vector $\mathbf{u}_k$

$$\begin{aligned}
\mathbf{u}_k &= -\left[\mathbf{R}_k + \mathbf{S}_k^T\mathbf{Q}_k\mathbf{S}_k\right]\mathbf{S}_k^T\mathbf{Q}_k\mathbf{C}_kx_k \\
&= \begin{bmatrix} -\left[R+B^TX_{k+1}B\right]^{-1}B^TX_{k+1}Ax_k \\ -\left[\mathbf{R}_{k+1}+\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{S}_{k+1}\right]\mathbf{S}_{k+1}^T\mathbf{Q}_{k+1}\mathbf{C}_{k+1}x_{k+1} \end{bmatrix} \\
&= \begin{bmatrix} u_k \\ \mathbf{u}_{k+1} \end{bmatrix}.
\end{aligned} \tag{2.85}$$

2.3.2 Closed form solution to the Riccati equation for optimal state estimation

In the previous subsection, the closed form solution of the Riccati equation for the LQ regulation problem is derived. In this section, the closed form solution of the state estimation problem is presented. This is based on the duality between the optimal control and optimal observer problem, see [13], instead of analyzing the specific problem again since it is sufficient for our purpose to obtain the solution. The stochastic system under consideration is now

$$\begin{aligned}
x_{k+1} &= Ax_k + Dw_k, \\
y_k &= Cx_k + v_k,
\end{aligned} \tag{2.86}$$

where the state is x, the output is y and where w and v are process and measurement noises satisfying

$$E\{w_iw_j^T\} = W, \quad E\{v_iv_j^T\} = V, \quad E\{v_iw_j^T\} = 0. \tag{2.87}$$

The problem is to find the Kalman gain matrix L_k of the filter

$$\hat{x}_{k+1} = A\hat{x}_k + L_k(y_k - C\hat{x}_k). \tag{2.88}$$

again minimizing the variance of the estimation error so that the following cost function is minimized [8]

$$J_k = \zeta_k^TE\{\tilde{x}_k\tilde{x}_k^T\}\zeta_k \quad \text{for any} \quad \zeta_k \in \mathbb{R}^n, \tag{2.89}$$

where $\tilde{x}$ denotes the estimation error. Due to the duality in the LQG problem, this problem can be treated as the optimal control problem of the dual system [13]

$$\begin{aligned}
\zeta_{j-1} &= A^T\zeta_j + C^Tu_j, \quad 1 \le j \le k, \\
\eta_j &= B^T\zeta_j, \quad 0 \le j \le k,
\end{aligned} \tag{2.90}$$

with the state feedback law for the input u_j chosen as

$$u_j = -L_{j-1}^T \zeta_j, \tag{2.91}$$

and the cost function to be minimized is defined by

$$J_k = \zeta_0^T Y_0 \zeta_0 + \sum_{j=1}^{k} (\eta_j^T W \eta_j + u_j^T V u_j), \quad Y_0 = E\{\tilde{x}_0 \tilde{x}_0^T\}. \tag{2.92}$$

Now the same algebra as in the preceding subsection can be used to obtain the closed-form solution

$$\begin{aligned} \mathbf{u}_0 &= -\mathbf{L}_0^T \zeta_0 = -\left(\mathbf{V}_0 + \mathbf{T}_0 \mathbf{W}_0 \mathbf{T}_0^T\right)^{-1} \mathbf{T}_0 \mathbf{W}_0 \mathbf{D}_0^T \zeta_0, \\ \mathbf{L}_0 &= \mathbf{D}_0 \mathbf{W}_0 \mathbf{T}_0 \left(\mathbf{V}_0 + \mathbf{T}_0 \mathbf{W}_0 \mathbf{T}_0^T\right)^{-1}. \end{aligned} \tag{2.93}$$

The closed-form solution of the Riccati equation becomes

$$Y_k = \mathbf{D}_k \mathbf{W}_k \mathbf{D}_k^T - \mathbf{D}_k \mathbf{W}_k \mathbf{T}_k \left(\mathbf{V}_k + \mathbf{T}_k \mathbf{W}_k \mathbf{T}_k^T\right)^{-1} \mathbf{T}_k \mathbf{W}_k \mathbf{D}_k^T, \tag{2.94}$$

which can be written using the matrix inversion lemma as

$$Y_k = \mathbf{D}_k \left(\mathbf{W}_k^{-1} + \mathbf{T}_k^T \mathbf{V}_k^{-1} \mathbf{T}_k\right)^{-1} \mathbf{D}_k^T, \tag{2.95}$$

where

$$\begin{aligned} \mathbf{D}_k &= \begin{bmatrix} D & AD & \dots & A^k D \end{bmatrix}, \\ \mathbf{T}_k &= \begin{bmatrix} 0 & CD & CAD & \dots & CA^{k-1}D \\ & 0 & CD & \dots & CA^{k-2}D \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & CD \\ & & & & 0 \end{bmatrix}, \\ \mathbf{W}_k &= \operatorname{diag}(W, W, \dots, W), \quad \mathbf{V}_k = \operatorname{diag}(V, V, \dots, V), \end{aligned} \tag{2.96}$$

and $\mathbf{W}_k$ and $\mathbf{V}_k$ contain $k+1$ diagonal blocks, respectively.

2.4 Suboptimal feedback and filtering

Thus far, optimal regulation and optimal estimation are considered. However, the closed-loop system and the observer, even with time-invariant system matrices, are time-varying since the optimal feedback (2.65) and observer (2.68) gains

$$\begin{aligned} K_k &= \left(R + B^T X_{k+1} B\right)^{-1} B^T X_{k+1} A, \\ L_k &= A Y_k C^T \left(V + C Y_k C^T\right)^{-1}, \end{aligned} \tag{2.97}$$

are time-varying. It is interesting to use suboptimal feedback and observer gains that do not actually minimize the performance indices but are constant so that

$$\begin{aligned} u_k &= -K \hat{x}_k, \\ \hat{x}_{k+1} &= A \hat{x}_k + B u_k + L(y_k - C \hat{x}_k). \end{aligned} \tag{2.98}$$

For these constant gains, the limit of the optimal gains as the final time N goes to infinity ($k \to -\infty$ and $k \to +\infty$, respectively) might be considered. The sequences X_k and Y_k can have several types of behavior: they can converge to steady-state matrices X_∞ and Y_∞ which may be zero, positive semidefinite or positive definite. They can also fail to converge to finite matrices.

If they do converge, then for large negative k evidently $X \triangleq X_k = X_{k+1}$ and for large positive k, $Y \triangleq Y_k = Y_{k+1}$. Thus, in the limit, (2.66) and (2.69) become the algebraic Riccati equations (ARE's)

$$\begin{aligned} X &= C^T Q C + A^T X A - A^T X B \left(R + B^T X B\right)^{-1} B^T X A, \\ Y &= D W D^T + A Y A^T - A Y C^T \left(V + C Y C^T\right)^{-1} C Y A^T, \end{aligned} \tag{2.99}$$

in which no time-dependence is present. There are several ways to solve these ARE's, for example by iteration or non-recursive methods using the eigenvalues and -vectors of the Hamiltonians. The solutions can be used to compute the steady-state gains K_∞ and L_∞ which result in a time-invariant feedback and observer.

2.5 Tracking using LQG control

The principle of LQG control is normally used for regulator problems, i.e., stabilizing the states to the equilibrium. If the LQG algorithm is to be used for tracking purposes, some additional measures have to be taken. These are discussed in this section.

2.5.1 Tracking time-varying reference inputs

Feedforward can be used to track general inputs, but unfortunately this technique suffers from the lack of robustness. The integral control technique can be generalized to track reference inputs that are, for instance, bandpass, sinusoidal or ramps. Integral control works by increasing the loop gain in the frequency band of the reference input. The tracking error is to be filtered by an appropriate filter and the filter output is included within the cost function used in generating the LQG gain. Perfect tracking of sinusoidal reference inputs (with known frequency) can for example be obtained by appending an oscillator to the tracking error before generating the LQG gain. When the oscillator poles are located at the frequency of the reference an infinite loop gain is achieved which yields perfect tracking. This can be generalized for any reference by the internal model principle [16]:

Zero steady-state tracking error is obtained when the poles in the Laplace transform of the reference input are also poles of the loop transfer function.

In the next subsection, two examples of feedforward control and integral control for a constant reference signal are presented.

2.5.2 Tracking constant reference inputs

Two approaches used to track constant reference inputs are presented in this subsection. A feedforward control input can be applied to force the system to exhibit the desired behavior. Integral control, a simple form of the internal model principle, can also be applied to guarantee a zero steady-state tracking error.

Feedforward control

In this case, the control law

$$u_k = -K x_k + K_r r, \tag{2.100}$$

is used, where $K_r r$ is termed a feedforward control. If this control law is used in system (2.12), the following is obtained

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k u_k = A_k x_k + B_k \left(-K x_k + K_r r\right) = \left(A_k - B_k K\right) x_k + B K_r r, \\ y_k &= C x_k. \end{aligned} \tag{2.101}$$

If the closed-loop system is stable, the feedforward control term generates the following steady-state reference output ($x_{k+1} = x_k$ when $k \to \infty$)

$$\begin{aligned} x_\infty &= (A - BK)\, x_\infty + BK_r r, \\ (I - (A - BK))\, x_\infty &= BK_r r, \\ x_\infty &= (I - (A - BK))^{-1} BK_r r, \\ y_\infty &= Cx_\infty = C\,(I - (A - BK))^{-1} BK_r r, \end{aligned} \tag{2.102}$$

so in order to let the output track the desired reference, the feedforward gain K_r should be the inverse of the DC-gain

$$K_r = \Big(C\,(I - (A - BK))^{-1} B\Big)^{-1}. \tag{2.103}$$

The use of feedback often greatly reduces the sensitivity of the output to plant errors but feedforward does not share this property. Therefore, feedforward control should be used to achieve desired reference outputs only when the plant model is known to be accurate.

Integral control

Integral feedback is used to zero out steady-state errors when tracking constant signals. Integral control can be regarded as an example of the internal model principle and is generated in a LQG setting by appending an integrator to the plant before computation of the feedback gains. The integral of the error between the reference input and the reference output is generated as follows

$$e_{k+1} = e_{k-1} + e_k T = e_{k-1} + (r - Cx_k)T, \tag{2.104}$$

where T is the sample time of the discrete system. Note that (2.104) requires the integration of each of the reference outputs, making the number of integrators equal to the number of reference outputs. The augmented state model is the combination of the plant state equation (2.12) and the state equation for the integral of the error

$$\left[\begin{array}{c} x_{k+1} \\ \hline e_{k+1} \end{array}\right] = \left[\begin{array}{c|c} A & 0 \\ \hline -CT & I \end{array}\right] \left[\begin{array}{c} x_k \\ \hline e_k \end{array}\right] + \left[\begin{array}{c} B \\ \hline 0 \end{array}\right] u_k + \left[\begin{array}{c} 0 \\ \hline IT \end{array}\right] r. \tag{2.105}$$

Now the LQG theory can be used to design a state feedback for the augmented plant ignoring the constant reference input and using a cost function that penalizes the integral of the error

$$\begin{aligned} J &= \frac{1}{2} e_N^T Q_N e_N + \frac{1}{2} \sum_{k=0}^{N-1} \left(e_k^T Q_k e_k + u_k^T R_k u_k \right) \\ &= \frac{1}{2} \left[\begin{array}{c|c} x_N^T & e_N^T \end{array}\right] \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & Q \end{array}\right] \left[\begin{array}{c} x_N \\ \hline e_N \end{array}\right] + \frac{1}{2} \sum_{k=0}^{N-1} \left(\left[\begin{array}{c|c} x_k^T & e_k^T \end{array}\right] \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & Q \end{array}\right] \left[\begin{array}{c} x_k \\ \hline e_k \end{array}\right] + u_k^T R u_k \right). \end{aligned} \tag{2.106}$$

The optimal control is then

$$u_k = -K \left[\begin{array}{c} x_k \\ \hline e_k \end{array}\right] = \left[\begin{array}{c|c} K_x & K_i \end{array}\right] \left[\begin{array}{c} x_k \\ \hline e_k \end{array}\right], \tag{2.107}$$

or written in terms of the tracking error

$$u_k = -K_x x_k - K_i \sum_{k=0}^{N} (r - y_k)\, T, \tag{2.108}$$

which shows that the control includes integral feedback.

Chapter 3

Data-Based LQG control

In the previous chapter, the model based LQG regulator has been derived. This chapter deals with the data-based counterpart. It is shown that the optimal control problem together with the optimal Kalman filter can be computed using the Markov parameters of a system instead of its state-space representation. The Markov parameters equal the impulse response data of the system at the sampled times. For now, they are supposed to be known but they can directly be estimated from input-output data as is explained in chapter 4. In this chapter, the data-based control theory for LQG control, as stated in [20], will be presented. All mathematics are presented in more detail than in the paper.

3.1 Introduction

Consider the strictly proper linear time invariant discrete-time system

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Dw_k, \\ y_k &= Cx_k + v_k, \end{aligned} \tag{3.1}$$

where x_k is the n dimensional state vector, u_k is the m dimensional control input vector, y_k is the ℓ dimensional output vector of interest, w_k is the n_w dimensional vector of model uncertainties and v_k is the ℓ dimensional disturbance vector corrupting the output. Please note that the matrix D denotes the model uncertainty matrix and not the direct transmission matrix. Suppose the Markov parameters from the input u to the output y: $M_i = CA^{(i-1)}B, \quad i = 1, 2, \dots, N-1$, the Markov parameters from the disturbance w to the output y: $H_i = CA^{(i-1)}D, \quad i = 1, 2, \dots, N+1$, and the disturbance covariance matrices W and V are known. The databased LQG problem is to find the optimal control sequence:

$$u_k = f(M_i, H_j, W, V, Q, R, u_{k-1}, y_{k-1}) \qquad k = 0, 1, 2, \dots, N-1, \quad i = 1, 2, 3, \dots, N-1, \quad j = 1, 2, 3, \dots, N+1 \tag{3.2}$$

such that the quadratic cost function

$$J = \mathbf{E}\left\{ y_N^T Q y_N + \sum_{k=0}^{N-1} \left(y_k^T Q y_k + u_k^T R u_k \right) \right\}, \tag{3.3}$$

is minimized.

3.2 Data-based LQG theory

In the first part of this section, the optimal feedback gain based on only Markov parameters is computed. However, not the estimated state of the system is fed back but the estimated states pre-multiplied with some vector. For this reason, the *controller state vector* is introduced in the second part. The data-based LQG controller will be derived using the optimal control results and the closed-form solutions from section 2.3.

3.2.1 Data-based optimal feedback gain

The model based optimal control law (see section 2.3) is

$$u_k = -(R + B^T X_{k+1} B)^{-1} B^T X_{k+1} A \hat{x}_k, \tag{3.4}$$

where X_{k+1} is the solution of the difference Ricatti equation (2.66). To put this control law in terms of only Markov parameters, the closed-form solution to this Ricatti equation (2.79) is used

$$\begin{aligned} X_{k+1} &= \mathbf{C}_{k+1}^T (\mathbf{Q}_{k+1}^{-1} + \mathbf{S}_{k+1}\mathbf{R}_{k+1}^{-1}\mathbf{S}_{k+1}^T)^{-1}\mathbf{C}_{k+1}, \qquad k = 1, 2, \ldots, N-1, \\ &= \mathbf{C}_{k+1}^T \mathbf{K}_{k+1} \mathbf{C}_{k+1}, \qquad\qquad\qquad\qquad\quad k = 1, 2, \ldots, N-1, \end{aligned} \tag{3.5}$$

where

$$\begin{aligned} \mathbf{C}_{k+1} &= \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{N-k+1} \end{bmatrix}, \\ \mathbf{K}_{k+1} &= (\mathbf{Q}_{k+1}^{-1} + \mathbf{S}_{k+1}\mathbf{R}_{k+1}^{-1}\mathbf{S}_{k+1}^T)^{-1}, \end{aligned} \tag{3.6}$$

where

$$\begin{aligned} \mathbf{S}_{k+1} &= \begin{bmatrix} 0 & & & & \\ CB & 0 & & & \\ CAB & CB & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ CA^{N-k-2}B & CA^{N-k-1}B & \cdots & CB & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & & & & \\ M_1 & 0 & & & \\ M_2 & M_1 & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ M_{N-k-1} & M_{N-k-2} & \cdots & M_1 & 0 \end{bmatrix}, \qquad \mathbf{S}_N = 0, \end{aligned} \tag{3.7}$$

$$\mathbf{Q}_{k+1} = diag(Q, Q, \ldots, Q), \quad \mathbf{R}_{k+1} = diag(R, R, \ldots, R), \tag{3.8}$$

and $\mathbf{Q}_{k+1}$ and $\mathbf{R}_{k+1}$ contain $N-k$ diagonal blocks, respectively. Now substitute (3.5) in (3.4) to yield

$$\begin{aligned} u_k &= -(R + B^T\mathbf{C}_{k+1}^T\mathbf{K}_{k+1}\mathbf{C}_{k+1}B)^{-1}B^T\mathbf{C}_{k+1}^T\mathbf{K}_{k+1}\mathbf{C}_{k+1}A\hat{x}_k \\ &= -(R + \mathbf{B}_{k+1}^T\mathbf{K}_{k+1}\mathbf{B}_{k+1})^{-1}\mathbf{B}_{k+1}^T\mathbf{K}_{k+1} \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-k} \end{bmatrix} \hat{\mathbf{x}}_k, \end{aligned} \tag{3.9}$$

where

$$\mathbf{B}_{k+1} = \mathbf{C}_{k+1}B = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k-1} \end{bmatrix} B = \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{N-k} \end{bmatrix},$$

The optimal control problem can be summarized as follows

$$u_k = \mathbf{G}_k \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-k} \end{bmatrix} \hat{\mathbf{x}}_k, \tag{3.10}$$

where

$$\mathbf{G}_k = -(R + \mathbf{B}_{k+1}^T \mathbf{K}_{k+1} \mathbf{B}_{k+1})^{-1} \mathbf{B}_{k+1}^T \mathbf{K}_{k+1}, \tag{3.11}$$

$$\mathbf{B}_{k+1} = \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{N-k} \end{bmatrix}, \tag{3.12}$$

$$\mathbf{K}_{k+1} = (\mathbf{Q}_{k+1}^{-1} + \mathbf{S}_{k+1} \mathbf{R}_{k+1}^{-1} \mathbf{S}_{k+1}^T)^{-1}, \tag{3.13}$$

$$\mathbf{S}_{k+1} = \begin{bmatrix} 0 & & & & \\ M_1 & 0 & & & \\ M_2 & M_1 & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ M_{N-k-1} & M_{N-k-2} & \cdots & M_1 & 0 \end{bmatrix}, \qquad \mathbf{S}_N = 0, \tag{3.14}$$

$$\mathbf{Q}_{k+1} = diag(Q, Q, \ldots, Q), \quad \mathbf{R}_{k+1} = diag(R, R, \ldots, R), \tag{3.15}$$

and $\mathbf{Q}_{k+1}$ and $\mathbf{R}_{k+1}$ contain $N-k$ diagonal blocks, respectively. Now equations (3.10)-(3.15) state the optimal control problem in terms of the Markov parameters of the system and the vector

$$\begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-k} \end{bmatrix} \hat{\mathbf{x}}_k. \tag{3.16}$$

In equation (3.10), $\mathbf{G_k}$ can be interpreted as the optimal gain, used to feed back vector (3.16), computed using only Markov parameters of the system. To obtain the total data-based LQG controller, vector (3.16) has also to be expressed in Markov parameters, this is done in the next subsection.

3.2.2 Data-based optimal estimation

In the previous subsection, the optimal feedback gain based on the Markov parameters is presented. Now the vector (3.16) has to be expressed in terms of these parameters as well. For this purpose, a *controller state vector*, $\overline{x}_k$, is introduced which is defined by

$$\overline{x}_k \triangleq \mathbf{C}_k \hat{x}_k = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k} \end{bmatrix} \hat{x}_k, \tag{3.17}$$

where $\hat{x}_k$ is the optimal estimation of the plant states x_k (see section 2.2.2),

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_{k-1} + L_{k-1}(y_{k-1} - C\hat{x}_{k-1}), \tag{3.18}$$

where L_{k-1} is the estimator gain which will be computed later. Note that the dimension of this vector is time-varying. When $k = 0$, $\bar{x}_0$ is a $N\ell$ dimensional vector and when $k = N-1$, $\bar{x}_{N-1}$ is a 2ℓ dimensional vector. Now it is shown how to compute these controller states using only the Markov parameters:

$$\begin{aligned}
\overline{x}_k \triangleq \mathbf{C}_k\hat{x}_k &= \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k} \end{bmatrix}\hat{x}_k \\
&= \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-k+1} \end{bmatrix}\hat{x}_{k-1} + \begin{bmatrix} CB \\ CAB \\ \vdots \\ CA^{N-k}B \end{bmatrix}u_{k-1} \\
&\quad + \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k} \end{bmatrix}L_{k-1}(y_{k-1} - C\hat{x}_{k-1}) \\
&= \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-k+1} \end{bmatrix}\hat{x}_{k-1} + \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{N-k+1} \end{bmatrix}u_{k-1} + \mathbf{F}_k(y_{k-1} - C\hat{x}_{k-1}) \\
&= \begin{bmatrix} -\mathbf{F}_k & \mathbf{I}_{(N-k+1)n_y} \end{bmatrix}\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k+1} \end{bmatrix}\hat{x}_{k-1} + \mathbf{B}_k u_{k-1} + \mathbf{F}_k y_{k-1} \\
&= \mathbf{A}_k\overline{x}_{k-1} + \mathbf{B}_k u_{k-1} + \mathbf{F}_k y_{k-1},
\end{aligned} \tag{3.19}$$

where

$$\mathbf{A}_k \triangleq \begin{bmatrix} -\mathbf{F}_k & \mathbf{I}_{N-k+1}l \end{bmatrix}, \quad \mathbf{B}_k \triangleq \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{N-k+1} \end{bmatrix}, \quad \mathbf{F}_k \triangleq \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k} \end{bmatrix} L_{k-1}. \tag{3.20}$$

The matrix $\mathbf{F}_k$ from equation (3.20) has to be written in terms of the Markov parameters. In order to do so, first L_{k-1} has to be expressed in terms of the Markov parameters. This estimator gain is given by equation (2.68), where Y_{k-1} is the solution to the closed-form expression of the Ricatti equation (2.95). These are repeated here

$$L_{k-1} = AY_{k-1}C^T(V + CY_{k-1}C^T)^{-1}, \tag{3.21}$$

$$\begin{aligned}
Y_{k-1} &= \mathbf{D}_{k-1}(\mathbf{W}_{k-1}^{-1} + \mathbf{T}_{k-1}^T\mathbf{V}_{k-1}^{-1}\mathbf{T}_{k-1})^{-1}\mathbf{D}_{k-1}^T, \\
&= \mathbf{D}_{k-1}\mathbf{P}_k\mathbf{D}_{k-1}^T,
\end{aligned} \tag{3.22}$$

where

$$\mathbf{P}_k \triangleq (\mathbf{W}_{k-1}^{-1} + \mathbf{T}_{k-1}^T V_{k-1}^{-1}\mathbf{T}_{k-1})^{-1}, \tag{3.23}$$

$$\mathbf{D}_{k-1} = \begin{bmatrix} D & AD & \cdots & A^{k-1}D \end{bmatrix},$$

$$\mathbf{T}_{k-1} = \begin{bmatrix} 0 & CD & CAD & \cdots & CA^{k-2}D \\ & 0 & CD & \cdots & CA^{k-3}D \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & CD \\ & & & & 0 \end{bmatrix} = \begin{bmatrix} 0 & H_1 & H_2 & \cdots & H_{k-1} \\ & 0 & H_1 & \cdots & H_{k-2} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & H_1 \\ & & & & 0 \end{bmatrix}, \tag{3.24}$$

$$\mathbf{W}_{k-1} = diag\{W, \cdots, W, W\}, \mathbf{V}_{k-1} = diag\{V, \cdots, V, V\}, \tag{3.25}$$

and $\mathbf{W}_{k-1}$ and $\mathbf{V}_{k-1}$ contain k diagonal blocks. Use (3.22) in (3.21) and (3.21) in (3.20), to express $\mathbf{F}_k$ as

$$\begin{aligned}\mathbf{F}_k &= \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k} \end{bmatrix} L_{k-1} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{N-k} \end{bmatrix} AY_{k-1}C^T(V + CY_{k-1}C^T)^{-1} \\ &= \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-k+1} \end{bmatrix} \mathbf{D}_{k-1}\mathbf{P}_k\mathbf{D}_{k-1}^T C^T(V + C\mathbf{D}_{k-1}\mathbf{P}_k\mathbf{D}_{k-1}^T C^T)^{-1} \\ &= \mathbf{H}_k\mathbf{P}_k\mathbf{N}_k^T(V + \mathbf{N}_k\mathbf{P}_k\mathbf{N}_k^T)^{-1},\end{aligned} \tag{3.26}$$

where

$$\mathbf{N}_k \triangleq C\mathbf{D}_{k-1} = \begin{bmatrix} CD & CAD & \cdots & CA^{k-1}D \end{bmatrix} = \begin{bmatrix} H_1 & H_2 & \cdots & H_k \end{bmatrix}, \tag{3.27}$$

and

$$\begin{aligned}\mathbf{H}_k &\triangleq \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-k+1} \end{bmatrix} \mathbf{D}_{k-1} \\ &= \begin{bmatrix} CAD & CA^2D & \cdots & CA^kD \\ CA^2D & CA^3D & \cdots & CA^{k+1}D \\ \vdots & \vdots & \ddots & \cdots \\ CA^{N-k+1}D & CA^{N-k+2} & \cdots & CA^ND \end{bmatrix} \\ &= \begin{bmatrix} H_2 & H_3 & \cdots & H_{k+1} \\ H_3 & H_4 & \cdots & H_{k+2} \\ \vdots & \vdots & \ddots & \vdots \\ H_{N-k+2} & H_{N-k+3} & \cdots & H_{N+1} \end{bmatrix}.\end{aligned} \tag{3.28}$$

Now everything is expressed in Markov parameters, the result for the controller state (3.17) is summarized. The data-based controller state equation is given in terms of the Markov parameter sequence as

$$\begin{aligned}\overline{x}_k &= \mathbf{A}_k\overline{x}_{k-1} + \mathbf{B}_k u_{k-1} + \mathbf{F}_k y_{k-1}, \\ \overline{x}_0 &= 0,\end{aligned} \tag{3.29}$$

where $\mathbf{A}_k$, $\mathbf{B}_k$ and $\mathbf{F}_k$ are time-varying gain matrices and given by

$$\mathbf{A}_k = \begin{bmatrix} -\mathbf{F}_k & \mathbf{I}_{N-k+1}l \end{bmatrix}, \tag{3.30}$$

$$\mathbf{B}_k = \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{N-k+1} \end{bmatrix}, \tag{3.31}$$

$$\mathbf{F}_k = \mathbf{H}_k\mathbf{P}_k\mathbf{N}_k^T(V + \mathbf{N}_k\mathbf{P}_k\mathbf{N}_k^T)^{-1}, \tag{3.32}$$

$$\mathbf{H}_k = \begin{bmatrix} H_2 & H_3 & \cdots & H_{k+1} \\ H_3 & H_4 & \cdots & H_{k+2} \\ \vdots & \vdots & \ddots & \vdots \\ H_{N-k+2} & H_{N-k+3} & \cdots & H_{N+1} \end{bmatrix}, \tag{3.33}$$

$$\mathbf{P}_k = (\mathbf{W}_{k-1}^{-1} + \mathbf{T}_{k-1}^T \mathbf{V}_{k-1}^{-1} \mathbf{T}_{k-1})^{-1}, \tag{3.34}$$

$$\mathbf{N}_k = \begin{bmatrix} H_1 & H_2 & \cdots & H_k \end{bmatrix}, \tag{3.35}$$

$$\mathbf{T}_{k-1} = \begin{bmatrix} 0 & H_1 & H_2 & \cdots & H_{k-1} \\ & 0 & H_1 & \cdots & H_{k-2} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & H_1 \\ & & & & 0 \end{bmatrix}, \tag{3.36}$$

$$\mathbf{W}_{k-1} = diag\{W, \cdots, W, W\}, \quad \mathbf{V}_{k-1} = diag\{V, \cdots, V, V\}, \tag{3.37}$$

and $\mathbf{W}_{k-1}$ and $\mathbf{V}_{k-1}$ contain k diagonal blocks, respectively. Now equations (3.29) - (3.37) give the data-based controller states completely in terms of Markov parameter sequences. The controller state vector can be interpreted as the estimated output and the predicted estimated outputs over the next $N-k$ samples if no input and innovation signals are taken into account.
The optimal data-based LQG control law associated with the cost function (3.3) is now given by

$$u_k = \mathbf{G}_k \begin{bmatrix} \mathbf{0}_\ell & \mathbf{I}_{(N-k)\ell} \end{bmatrix} \overline{x}_k, \tag{3.38}$$

where $\mathbf{G}_k$ and $\overline{x}_k$ are given in (3.11) and (3.29), respectively. The advantage of this equation is the fact that the optimal input can be determined from only past values of the input and output and the Markov parameters of the system, which can also be determined using only past input/output data as is explained in the next chapter. For SISO systems, the number of Markov parameters required to completely define a proper n^{th} order system is $2n$, [1]. As long as $N < 2n-1$, the data-based algorithm needs less information to compute the optimal input than is needed to construct a model and use the model-based algorithm. The main disadvantage of the data-based control law is the fact that the matrices will become very large if the horizon is extended to large values. The need for these large matrices arise due to the use of the closed form solutions of the Riccati equations.
The controller can be implemented using the recursive form given in (3.29) and (3.38) or in a batch form, see Skelton ([20]). The data-based LQG controller only uses the system Markov parameters and past values of the input and output of the system. In the next chapter the computation of the Markov parameters using input/output data is discussed.

Chapter 4

Markov Parameters

In order to use the data-based LQG algorithm presented in chapter 3, the Markov parameters from the system under consideration are needed. Hence, a method to obtain these parameters with the use of Auto-Regressive Models with explicit Markov parameter coefficients (ARMarkov) is presented. Since Markov parameters equal the impuls response of a system, they represent the system in a certain way. Although the algorithm to estimate these parameters makes use of models, the resulting controller is still regarded as data-based since this estimation process is incorporated in the overall algorithm and no separate model identification steps are necessary.

4.1 Introduction

Dynamical systems can be represented in many forms, for instance with state-space descriptions, transfer functions and input-output models. Another way to represent a system is by its impulse response behavior. For discrete systems, the impulse response at the sample times are called the Markov parameters (M_i) of a system and can be computed from the state-space model by

$$M_i = CA^{i-1}B. \tag{4.1}$$

Another way to obtain these parameters is to directly measure the impulse response. Obviously this is not a very useful method for many systems. It is also possible to compute the Markov parameters from any sufficiently exciting sequence of input-output data. This can be done by the use of ARMarkov models and is explained in this chapter. Also some simulations are performed.

4.2 The ARMarkov model

In this chapter the more general n^{th} order, m-input, ℓ-output linear time invariant discrete-time bi-proper model in state-space format

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \\ y(k) &= Cx(k) + Du(k), \end{aligned} \tag{4.2}$$

is considered, the step towards the strictly proper system used in the data-based algorithm is easily made by setting the direct transmission matrix D equal to the null matrix. If the states in this system are repeatedly substituted, then for some $p \geq 0$,

$$\begin{aligned} x(k+p) &= A^p x(k) + \mathbf{C}_p u_p(k), \\ y_p(k) &= \mathbf{O}_p x(k) + \mathbf{T}_p u_p(k), \end{aligned} \tag{4.3}$$

where u_p and y_p are vectors containing input and output data going p steps into the future starting with $u(k)$ and $y(k)$,

$$u_p(k) = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+p-1) \end{bmatrix}, \quad y_p(k) = \begin{bmatrix} y(k) \\ y(k+1) \\ \vdots \\ y(k+p-1) \end{bmatrix}. \tag{4.4}$$

The matrix $\mathbf{C}_p$ in equation (4.3) is an $n \times pm$ controllability matrix, $\mathbf{O}_p$ a $p\ell \times n$ observability matrix and $\mathbf{T}_p$ a $p\ell \times pm$ Toeplitz matrix of the system Markov parameters,

$$\mathbf{C}_p = \begin{bmatrix} A^{p-1}B & \cdots & AB & B \end{bmatrix}, \quad \mathbf{O}_p = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{p-1} \end{bmatrix},$$

$$\mathbf{T}_p = \begin{bmatrix} D & 0 & \cdots & \cdots & 0 \\ CB & D & \ddots & \ddots & \vdots \\ CAB & CB & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ CA^{p-2}B & \cdots & \cdots & CB & D \end{bmatrix}. \tag{4.5}$$

As long as $p\ell \geq n$, it is guaranteed for an observable system that there exists a matrix $\mathbf{M}$ such that (see [19])

$$A^p + \mathbf{M}\mathbf{O}_p = 0 \tag{4.6}$$

The existence of $\mathbf{M}$ ensures that, for $k \geq 0$, an expression exists for $x(k+p)$ where the state variable is completely eliminated from equation (4.3)

$$\begin{aligned}
&A^p + \mathbf{M}\mathbf{O}_p = 0 \Rightarrow A^p = -\mathbf{M}\mathbf{O}_p, \\
&x(k+p) = -\mathbf{M}\mathbf{O}_p x(k) + \mathbf{C}_p u_p(k), \\
&y_p(k) = \mathbf{O}_p x(k) + \mathbf{T}_p u_p(k) \Rightarrow \mathbf{M}y_p(k) = \mathbf{M}\mathbf{O}_p x(k) + \mathbf{M}\mathbf{T}_p u_p(k) \Rightarrow, \\
&\mathbf{M}\mathbf{O}_p x(k) = \mathbf{M}y_p(k) - \mathbf{M}\mathbf{T}_p u_p(k), \\
&x(k+p) = -\mathbf{M}y_p(k) + \mathbf{M}\mathbf{T}_p u_p(k) + \mathbf{C}_p u_p(k),
\end{aligned}$$

so,

$$x(k+p) = (\mathbf{C}_p + \mathbf{M}\mathbf{T}_p)u_p(k) - \mathbf{M}y_p(k). \tag{4.7}$$

Equation (4.7) is independent of the systems stability and in order to solve it, the initial condition $x(0)$ is not required. Of course, the following also holds

$$x(k+q) = A^q x(k) + \mathbf{C}_q u_q(k). \tag{4.8}$$

Now go p time steps back in equation (4.7) and then substitute this $x(k)$ in equation (4.8) to obtain, for $k \geq p$, an expression for the state $x(k+q)$

$$x(k+q) = A^q(\mathbf{C}_p + \mathbf{M}\mathbf{T}_p)u_p(k-p) - A^q\mathbf{M}y_p(k-p) + \mathbf{C}_q u_q. \tag{4.9}$$

Finally, if equation (4.9) is substituted in $y_p(k+q) = \mathbf{O}_p x(k+q) + \mathbf{T}_p u(k+q)$, an input/output model representation is obtained

$$y_p(k+q) = \mathbf{O}_p A^q(\mathbf{C}_p + \mathbf{M}\mathbf{T}_p)u_p(k-p) - \mathbf{O}_p A^q\mathbf{M}y_p(k-p) + \mathbf{H}u_q(k) + \mathbf{T}_p u_p(k+q), \tag{4.10}$$

where $\mathbf{H}$ is only defined when $q > 0$ and has the following structure

$$\mathbf{H} = \mathbf{O}_p\mathbf{C}_q = \begin{bmatrix} CA^{q-1}B & CA^{q-2}B & \cdots & CB \\ CA^{q}B & CA^{q-1}B & \cdots & CAB \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p+q-2}B & CA^{p+q-3}B & \cdots & CA^{p-1}B \end{bmatrix}. \tag{4.11}$$

Since $\mathbf{H}$ is called a Hankel matrix and since $\mathbf{T}_p$ is often referred to as a Toeplitz matrix, equation (4.10) is called a Hankel-Toeplitz model. The expression for $y(k+q)$ only involves past p input values from $u(k-p)$ to $u(k-1)$, past p output values from $y(k-p)$ to $y(k-1)$, the present input $u(k)$ and $p+q$ future inputs, see figure 4.1.

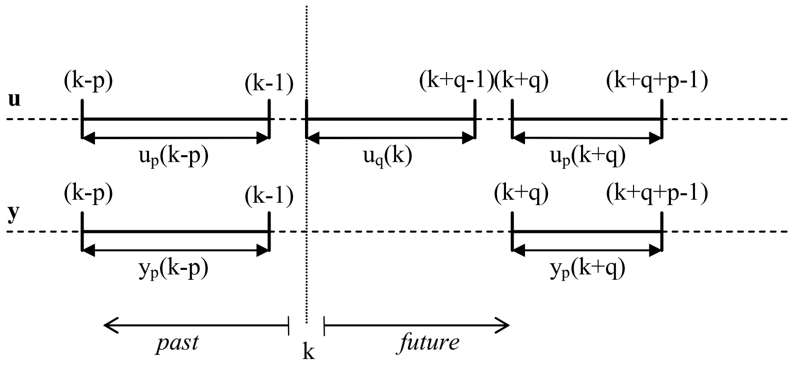


Figure 4.1: Illustration of the data partitioning

Equation (4.10) can be written in the general form

$$y_p(k+q) = \sum_{i=1}^{p} \alpha_i y(k-i) + \sum_{i=1}^{p} \beta_i u(k-i) + \sum_{i=0}^{q} \mathbf{H}u(k+i) + \sum_{i=0}^{q+p} \mathbf{T}_p u(k+i), \tag{4.12}$$

where the coefficients are related to the state space model A, B, C, D by

$$\begin{aligned} \begin{bmatrix} \alpha_p & \dots & \alpha_2 & \alpha_1 \end{bmatrix} &= -\mathbf{O}_p A^q \mathbf{M} \triangleq P_1, \\ \begin{bmatrix} \beta_p & \dots & \beta_2 & \beta_1 \end{bmatrix} &= \mathbf{O}_p A^q(\mathbf{C}_p + \mathbf{M}\mathbf{T}_p) \triangleq P_2. \end{aligned} \tag{4.13}$$

These models are all ARMarkov models. Consider the case when $q = 0$, then

$$y_p(k) = \begin{bmatrix} P_2 & P_1 & \mathbf{T}_p \end{bmatrix} \begin{bmatrix} u_p(k-p) \\ y_p(k-p) \\ u_p(k) \end{bmatrix}, \tag{4.14}$$

and the first element of these models is the Auto-Regressive moving average model with eXogenous input (ARX). The collection of models (4.14) can be used to directly identify the matrix $\mathbf{T}_p$ from a set of input-output data along with $P_1 = -\mathbf{O}_p\mathbf{M}$ and $P_2 = \mathbf{O}_p(\mathbf{C}_p + \mathbf{M}\mathbf{T}_p)$ as follows,

$$\begin{bmatrix} P_2 & P_1 & \mathbf{T}_p \end{bmatrix} \mathbf{V} = \mathbf{Y}, \tag{4.15}$$

where

$$\mathbf{Y} = \begin{bmatrix} y_p(k+p) & y_p(k+p+1) & \cdots & y_p(k+p+L) \end{bmatrix}, \tag{4.16}$$

$$\mathbf{V} = \begin{bmatrix} u_p(k) & u_p(k+1) & \cdots & u_p(k+L) \\ y_p(k) & y_p(k+1) & \cdots & y_p(k+L) \\ u_p(k+p) & u_p(k+p+1) & \cdots & u_p(k+p+L) \end{bmatrix}. \tag{4.17}$$

These are essentially the same equations as (4.12) where every column of $\mathbf{Y}$ and $\mathbf{V}$ coincides with a following time-step. The L denotes the total length of the available data. As can be seen there is a lot of duplication in the used data. This indicates that it is not a very efficient way to estimate these coefficients.
In order to identify the Markov parameters uniquely, the data set must be sufficiently long and rich so that the rows of $\mathbf{V}$ associated with the input data, u_p, are linearly independent. Since the width of the matrix $\mathbf{V}$ is $3p$, the length L should be at least $3p$, otherwise the matrix will never have full rank. To increase robustness against noise, L can be chosen larger since it then will be better filtered out.
A Hankel matrix $\mathbf{H}_0 = \mathbf{O}_p\mathbf{C}_p$ can now be obtained through

$$\mathbf{H}_0 = P_2 + P_1\mathbf{T}_p, \tag{4.18}$$

and from this matrix the first $2p-1$ Markov parameters CA^kB,
$k = 0, 1, \ldots, 2p-2$ can be extracted. If necessary, the following Markov parameters can be obtained by recognizing that $A^p = -\mathbf{MO}_p$ (see equation (4.6)),

$$\mathbf{H}_i = \mathbf{O}_p(A^p)^i\mathbf{C}_p = \mathbf{O}_p \underbrace{(-\mathbf{MO}_p)\ldots(-\mathbf{MO}_p)}_{i \quad times} \mathbf{C}_p = (-\mathbf{O}_p\mathbf{M})\ldots(-\mathbf{O}_p\mathbf{M})\mathbf{O}_p\mathbf{C}_p = (P_1)^i\mathbf{H}_0. \tag{4.19}$$

4.3 Simulation

In the previous section, it is shown how the Markov parameters of a system can be computed from input/output data. This section provides a simulation example. The model used for these simulations is a linear time invariant discrete-time system and can be represented by a double integrator with a resonance, as depicted in figure 4.2. The explicit state space description can be found in appendix B.

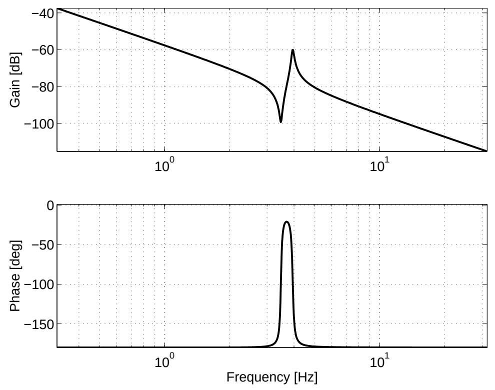
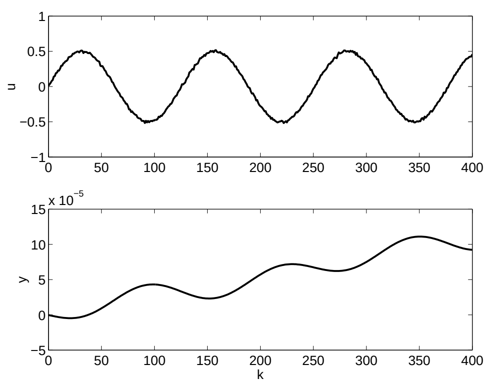


Figure 4.2: Simulation system

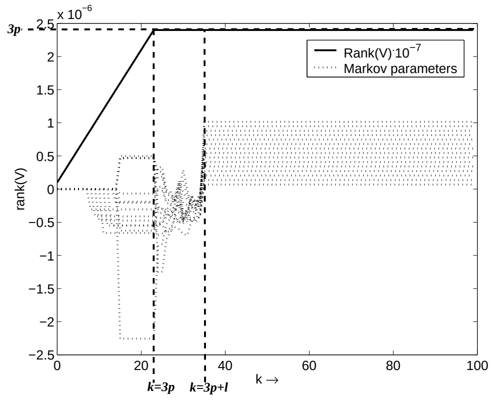
The algorithm from the previous section will be used to estimate the Markov parameters. It is implemented in Simulink so online estimation of the Markov parameters is possible. In order to solve the system of linear equations (4.15), the pseudo-inverse of $\mathbf{V}$ is used

$$\begin{bmatrix} P_2 & P_1 & \mathbf{T}_p \end{bmatrix} = \mathbf{YV}^{\dagger}, \tag{4.20}$$

where the superscript † denotes the pseudo-inverse (see appendix A.3). The input to the model is a sinusoid with additional white noise. Both the input and output of the model are measured and used in the algorithm, these are depicted in figure 4.3(a). In this simulation, the sample frequency is 1000 Hz and $p = 8$, this results in the estimation of $2 \cdot 8 - 1 = 15$ Markov parameters. The value for L should be at least $3p$ and to be on the safe side it is chosen to be $L = 27$. In figure 4.3(b) the first 15 estimated Markov parameters are depicted, together with the value of the rank of the matrix $\mathbf{V}$ (this starts at 1 since there is also an initial condition present and it is multiplied by a factor 10^{-7} for plotting purposes). As can be seen, the maximum rank ($3p = 24$) is reached after 24 samples, but only after $p + L = 35$ samples, the matrix $\mathbf{V}$ is fully filled and the Markov parameters are estimated correctly. This can be prevented by the use of another implementation where the size of this matrix is not static but adapts itself to the readily obtained data. Due to the possible rise of numerical problem when computing the rank of $\mathbf{V}$, it may be observed that the rank will decrease after some time.



(a) Input (u) and output (y) to the model



(b) Estimated Markov parameters together with rank($\mathbf{V}$)

Figure 4.3: In- and output to the model and the estimated Markov parameters

The estimated values correspond with the computed ones:

$$CA^iB = 10^{-5} \cdot \begin{bmatrix} 0.0068 & 0.0136 & 0.0205 & \cdots & 0.1018 \end{bmatrix}, \quad i = 0, 1, 2, \ldots, 14. \tag{4.21}$$

When these parameters are used for control purposes, it is of course important to know what dynamics they represent. The frequencies which are visible in the signal are determined by the sample time and the number of samples that are used to extract the information. The maximum frequency that can be extracted from a signal is dictated by the sample frequency (f_s) and is equal to the Nyquist frequency due to aliasing [17]

$$f_{max} = \frac{f_s}{2}. \tag{4.22}$$

The minimum frequency that can be extracted is dictated by the number of samples. It is the frequency of the lowest harmonic function that 'fits' in the amount of samples and therefore equal to

$$f_{min} = \frac{f_s}{N_s}, \tag{4.23}$$

where N_s denotes the number of samples over which the data is collected. These two frequencies are visualized in figure 4.4.

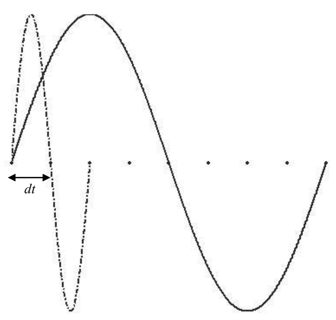


Figure 4.4: Maximum (dashed) and minimum (solid) frequency that can be extracted from data

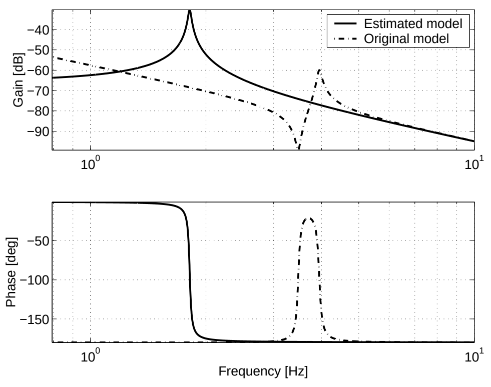
This is only a rule of thumb and can be used to determine the sample frequency and the number of samples. To check if the original system from figure 4.2 can be reconstructed from the estimated Markov parameters, a model is made from these parameters. First, the Markov parameters are used to build the Hankel matrices $\mathbf{H}_0$ and $\mathbf{H}_1$:

$$\mathbf{H}_0 = \begin{bmatrix} CB & \cdots & CA^nB \\ \vdots & \vdots & \vdots \\ CA^nB & \cdots & CA^{2n} \end{bmatrix}, \qquad \mathbf{H}_1 = \begin{bmatrix} CAB & \cdots & CA^{n+1}B \\ \vdots & \vdots & \vdots \\ CA^{n+1}B & \cdots & CA^{2n+1} \end{bmatrix}. \tag{4.24}$$

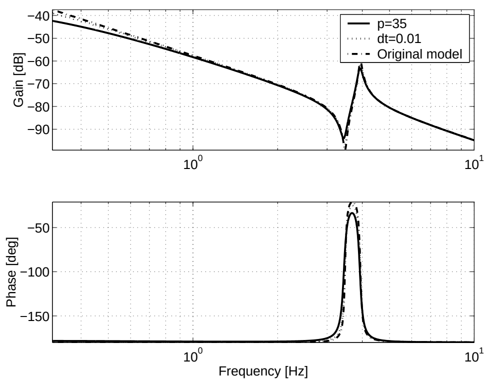
After that, an s^{th} order state-space model with m inputs and ℓ outputs is made by the following relations:

$$\begin{aligned} \overline{A} &= \Sigma_s^{-1/2} U_s^T \mathbf{H}_1 V_s \Sigma_s^{-1/2}, \\ \overline{B} &= \Sigma_s^{1/2} V_s^T \qquad \text{(first } m \text{ columns)}, \\ \overline{C} &= U_s \Sigma_s^{1/2} \qquad \text{(first } \ell \text{ rows)}, \\ \overline{D} &= D, \end{aligned} \tag{4.25}$$

where the matrices U_s and V_s are made up of s singular vectors of $\mathbf{H}_0$ and the diagonal matrix Σ_s is made up of the s corresponding singular values. As can be seen, to generate a model of order n, the largest Markov parameter used in $\mathbf{H}_1$ is $2n+1$. The first 15 Markov parameters obtained by sampling at 0.001 second and $p = 8$ samples are used. The original and realized system are depicted in figure 4.5(a). As can be seen, the estimated model does not correspond with the original one, except for high frequencies. To obtain better results, the number of used samples could be increased or the sample time could be decreased. In figure 4.5(b), these two things are applied ($dt = 0.001, \quad p = 35$ and $dt = 0.01, \quad p = 8$) and the estimated models together with the original model are depicted. As can be seen, the results are much better.



(a) $dt = 0.001$ and $p = 8$



(b) $dt = 0.001$, $p = 35$ and $dt = 0.01$, $p = 8$

Figure 4.5: Original system and realized systems with estimated Markov parameters

Chapter 5

Implementation and simulations of the LQG controller

In this chapter the implementation of the data-based LQG controller and the Markov estimation algorithm is provided. Some regulation simulations are presented to illustrate the equivalence of the data-based LQG controller and its model-based counterpart. Since discrete LQG control is done over a finite horizon, a moving horizon algorithm has to be developed to be able to control over infinite time. The implementation of this method and some drawbacks are discussed. As input to the data-based controller, various signals can be used, so it should be possible to obtain a tracking controller by using the error as one of the inputs to the algorithm instead of the systems output. This is done at the end of this chapter.

5.1 Implementation of the algorithms

In this section, the implementation of the LQG and Markov parameter estimation algorithms are presented. In order to use the LQG data-based algorithm online, the matrices derived in chapter 3 have to be filled and processed online in the Simulink environment. In order to do so, specific S-functions have been written in C which generate the matrices and some functions, that can perform linear algebra, for instance matrix multiplication, matrix inversion, transpose of the matrix and computation of the determinant. A major problem in the implementation of the algorithm, is the fact that the sizes of the matrices are varying during the simulation. Contact with The MathWorks made clear that it is not possible to generate such matrices in Simulink. Therefore another way has been chosen. In the C-functions which generate the various matrices, memory is reserved for the maximum number of elements of that specific matrix which depends on the control horizon (N). The dimensions and the maximum number of elements for each matrix are given in table 5.1.

Matrix	Size	Max. number of elements
H_k	$(N-k)$ x $(k+1)$	$floor(\frac{1}{2}N+\frac{1}{2})^2$
N_k	1 x $(k+1)$	$N-1$
T_k	$(k+1)$ x $(k+1)$	$(N-1)^2$
B_k	$(N-k)$ x 1	N
B_{k+1}	$(N-k-1)$ x 1	$(N-1)$
S_{k+1}	$(N-k-1)$ x $(N-k-1)$	$(N-1)^2$
W_k	$(k+1)$ x $(k+1)$	$(N-1)^2$
V_k	$(k+1)$ x $(k+1)$	$(N-1)^2$
Q_{k+1}	$(N-k-1)$ x $(N-k-1)$	$(N-1)^2$
R_{k+1}	$(N-k-1)$ x $(N-k-1)$	$(N-1)^2$

Table 5.1: The various matrices with their dimensions and maximum number of elements, $k = 0, 1, \ldots, N-1$.

This way, it is possible to write S-functions which fill the matrices in the correct manner. In order to accommodate matrix multiplications with matrices which vary in size during simulation, every multiplication function is specifically adjusted to the matrices which are multiplied. All S-functions are programmed in a general way, so various values for the tuning parameters Q, R, W, V, N can be used. Now the data-based LQG algorithm can be implemented online.
The algorithm to estimate the Markov parameters online is implemented in a similar way. Some S-functions have been developed to generate the used matrices and vectors. To perform operations like computing the pseudo-inverse and the singular value decomposition, the DSP blockset provided by Simulink has been used. Since this implementation is not very efficient, it would be better to use other techniques to solve the systems of linear equations. The connection from the computer to the actual experimental setup is made by WinCon, a program with which Simulink schemes can be used for real-time experiments. In order to compile the schemes, the target file for WinCon has to be rewritten. All functions and files are provided in the disc accompanying this report.
The following section discusses the implementation of the model-based and data-based moving horizon controllers. The idea is to restart the control algorithms every horizon (i.e., at N, $2N$,...,iN, etc.) as depicted in figure 5.1. A setup where the horizon shifts every sample is not possible in the data-based setup. The Markov parameters are held constant during every control horizon (i.e., during 0-N, N-$2N$, etc.), so the control action is based on a time-invariant model during one horizon. However, it is possible that the control action in the next horizon is based on another set of Markov parameters and thus on another model. This way, the controller is able to adapt to possible changes in the dynamics of the system and a time-varying controller is obtained.

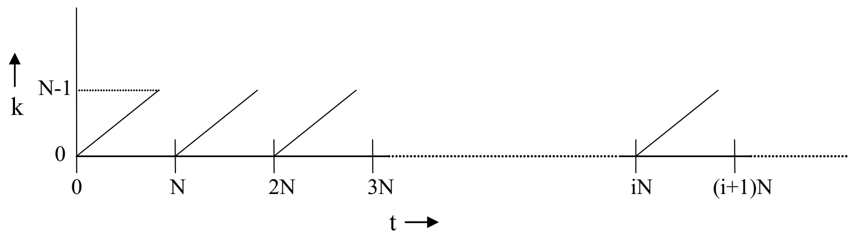


Figure 5.1: Illustration of time-line divided in horizons.

5.2 Moving horizon control

The solution to the discrete time LQG regulator problem is a finite horizon controller. This means that the control action is defined for only this finite period of time. In order to extend this period, a moving horizon scheme has to be developed. In this section, this is done for both the model and data-based LQG controller.

5.2.1 Model based Moving horizon controller

As is stated in section 2.3, the optimal input is given by

$$u_k = -K_k \hat{x}_k, \quad k = 0, 1, 2, \ldots, N-1, \tag{5.1}$$

where

$$K_k = \left(R + B^T X_{k+1} B\right)^{-1} B^T X_{k+1} A, \tag{5.2}$$

where X_{k+1} is the solution of the (backward) difference Riccati equation

$$\begin{aligned} X_k &= C^T Q C + A^T X_{k+1} A - A^T X_{k+1} B \left(R + B^T X_{k+1} B\right)^{-1} B^T X_{k+1} A, \\ X_N &= C^T Q C. \end{aligned} \tag{5.3}$$

The optimal state estimation, $\hat{x}_k$ can be obtained from

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L_k(y_k - C\hat{x}_k), \tag{5.4}$$

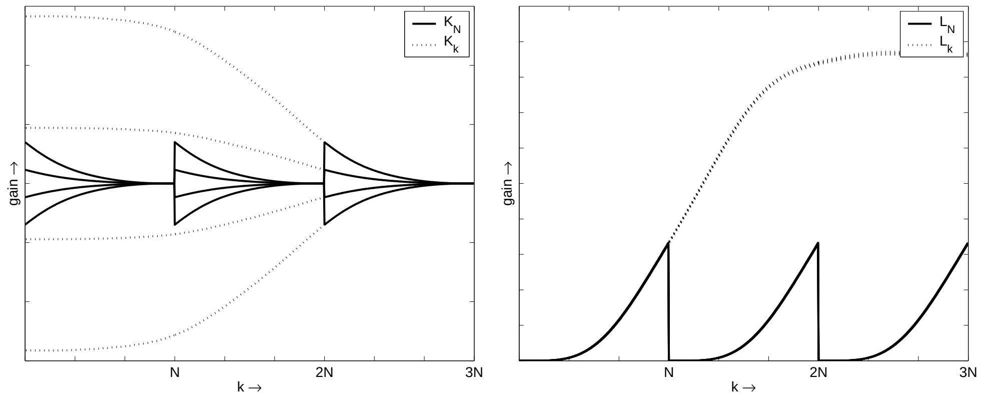
where the estimator gain L_k is given by

$$L_k = AY_kC^T \left(V + CY_kC^T\right)^{-1}, \tag{5.5}$$

where Y_k is the solution of the (forward) difference Riccati equation

$$\begin{aligned} Y_{k+1} &= DWD^T + AY_kA^T - AY_kC^T \left(V + CY_kC^T\right)^{-1} CY_kA^T, \\ Y_0 &= DWD^T. \end{aligned} \tag{5.6}$$

There are basically two approaches to restyle this controller into a moving horizon controller. The first one is to use the steady-state solutions of the difference Ricatti equations, which leads to the time-invariant suboptimal regulator with gains K_∞ and L_∞ (see section 2.4). However, if optimal control is desired, the computed gains K_k and L_k can be used each horizon. It is not possible to solve both difference Ricatti equations online, since equation (5.4) requires a backward recursion. To initialize the state estimator every next horizon, the last estimation from the preceding horizon can be used ($\hat{x}_{0_{i+1}} = \hat{x}_{N_i}$, where the second subscript denotes the horizon number). The problem when using the last setup, is that the optimal gains could be in their transient fase if the horizon (N) is small, as depicted in figure 5.2. The dotted lines show the gains if the horizon would be $3N$ while the solid lines show the actually used gains with the moving horizon scheme with horizon length N. As can be seen, they are still in the transient fase. The shorter the horizon is chosen, the lower the gains will be. To make this approach practical, there is a lower limit on N.



(a) Optimal feedback gain K_k and moving horizon feedback gain K_N

(b) Optimal estimator gain L_k and moving horizon estimator gain L_N

Figure 5.2: The moving horizon optimal gains could be in their transient fase if the horizon is small.

As can be seen in the figures, the optimal gains reach their steady-state value after some time, these values correspond to K_∞ and L_∞. Note that the steady-state value of the feedback gain correspond with $k = 0$, this is because of the backward recursion which is involved.

5.2.2 Data-based moving horizon controller

Since in the data-based control algorithm the closed-form solution to the Ricatti equation (see section 2.3.1) is used, a huge number of Markov parameters is needed to compute the steady-state solution. A lot of computational effort is needed to obtain all those parameters and to perform the operations on the large resulting matrices. A possible solution is to use other techniques to compute the linear algebra needed in the recursive implementation. However, the only applicable option in a short time is to use the optimal gains together with a small horizon. The Markov parameters are estimated every sample so it is possible to use these estimates every sample, this way the parameters are also updated during a horizon. The dynamics of the system are not expected to change very fast so the latest estimated parameters are held constant every horizon which results in a time-varying controller over the horizons. In order to initialize the data-based estimation process every horizon, it would be best to use the last value of the estimated controller vector

$$\overline{x}_{0_{i+1}} = \overline{x}_{N_i} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^N \end{bmatrix} \hat{x}_{N_i}, \tag{5.7}$$

where the subscript i denotes the horizon number (see figure 5.1). Unfortunately, this vector is not available straightforward since the controller state vector (3.17) is decreasing in dimension every sample during one horizon since this is inherent to the algorithm. However, this vector can be computed as follows. When model based optimal control is used, the following sequence is obtained for the estimated states

$$\begin{bmatrix} \hat{x}_{1_i} \\ \hat{x}_{2_i} \\ \hat{x}_{3_i} \\ \vdots \\ \hat{x}_{N_i} \end{bmatrix} = \begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^N \end{bmatrix} \hat{x}_{0_i} + \begin{bmatrix} B & 0 & 0 & \cdots & 0 \\ AB & B & 0 & \cdots & 0 \\ A^2B & AB & B & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \cdots & B \end{bmatrix} \begin{bmatrix} u_{0_i} \\ u_{1_i} \\ u_{2_i} \\ \vdots \\ u_{(N-1)_i} \end{bmatrix}$$
$$+ \begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ A & I & 0 & \cdots & 0 \\ A^2 & A & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{N-1} & A^{N-2} & A^{N-3} & \cdots & I \end{bmatrix} \begin{bmatrix} L_k & 0 & 0 & \cdots & 0 \\ 0 & L_{k+1} & 0 & \cdots & 0 \\ 0 & 0 & L_{k+2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & L_{k+N-1} \end{bmatrix} \begin{bmatrix} y_{0_i} - C\hat{x}_{0_i} \\ y_{1_i} - C\hat{x}_{1_i} \\ y_{2_i} - C\hat{x}_{2_i} \\ \vdots \\ y_{(N-1)_i} - C\hat{x}_{(N-1)_i} \end{bmatrix}. \tag{5.8}$$

If the last element of these estimated states, $\hat{x}_{N_i}$, is substituted in equation (5.7), the following value is obtained for the controller state vector

$$\overline{x}_{N_i} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} \hat{x}_{N_i}$$
$$= \begin{bmatrix} CA^N \\ CA^{N+1} \\ CA^{N+2} \\ \vdots \\ CA^{2N} \end{bmatrix} \hat{x}_{0_i} + \begin{bmatrix} CA^{N-1}B & CA^{N-2}B & \cdots & CB \\ CA^NB & CA^{N-1}B & \cdots & CAB \\ CA^{N+1}B & CA^NB & \cdots & CA^2B \\ \vdots & \vdots & \ddots & \vdots \\ CA^{2N-1}B & CA^{2N-2}B & \cdots & CA^NB \end{bmatrix} \begin{bmatrix} u_{0_i} \\ u_{1_i} \\ u_{2_i} \\ \vdots \\ u_{(N-1)_i} \end{bmatrix}$$
$$+ \begin{bmatrix} CA^{N-1} & CA^{N-2} & CA^{N-3} & \cdots & C \\ CA^N & CA^{N-1} & CA^{N-2} & \cdots & CA \\ CA^{N+1} & CA^N & CA^{N-1} & \cdots & CA^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{2N-1} & CA^{2N-2} & CA^{2N-3} & \cdots & CA^N \end{bmatrix} \begin{bmatrix} L_0 & 0 & 0 & \cdots & 0 \\ 0 & L_1 & 0 & \cdots & 0 \\ 0 & 0 & L_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & L_{N-1} \end{bmatrix} \begin{bmatrix} y_{0_i} - C\hat{x}_{0_i} \\ y_{1_i} - C\hat{x}_{1_i} \\ y_{2_i} - C\hat{x}_{2_i} \\ \vdots \\ y_{(N-1)_i} - C\hat{x}_{(N-1)_i} \end{bmatrix}$$

$$= \begin{bmatrix} CA^N \\ CA^{N+1} \\ CA^{N+2} \\ \vdots \\ CA^{2N} \end{bmatrix} \hat{x}_{0_i} + \begin{bmatrix} M_N & M_{N-1} & \cdots & M_1 \\ M_{N+1} & M_N & \cdots & M_2 \\ M_{N+2} & M_{N+1} & \cdots & M_3 \\ \vdots & \vdots & \ddots & \vdots \\ M_{2N} & M_{2N-1} & \cdots & M_{N+1} \end{bmatrix} \begin{bmatrix} u_{0_i} \\ u_{1_i} \\ u_{2_i} \\ \vdots \\ u_{(N-1)_i} \end{bmatrix}$$
$$+ \begin{bmatrix} \mathbf{Z}_0 & \mathbf{Z}_1 & \mathbf{Z}_2 & \cdots & \mathbf{Z}_{N-1} \end{bmatrix} \begin{bmatrix} y_{0_i} \\ y_{1_i} \\ y_{2_i} \\ \vdots \\ y_{(N-1)_i} \end{bmatrix} \tag{5.9}$$
$$- \begin{bmatrix} \mathbf{Z}_0 & \mathbf{Z}_1 & \mathbf{Z}_2 & \cdots & \mathbf{Z}_{N-1} \end{bmatrix} \left(\begin{bmatrix} 1 & \mathbf{0}_{(1,N)} \end{bmatrix} \begin{bmatrix} \overline{x}_{0_i} & \overline{x}_{1_i} & \cdots & \overline{x}_{(N-1)_i} \end{bmatrix} \right)^T,$$

where

$$\mathbf{Z}_k = \mathbf{E}_k \mathbf{P}_{k+1} \mathbf{N}_{k+1}^T (V + \mathbf{N}_{k+1} \mathbf{P}_{k+1} \mathbf{N}_{k+1}^T)^{-1}$$
$$\mathbf{E}_k = \begin{bmatrix} CA^{N-k}D & CA^{N-k+1}D & \cdots & CA^{N}D \\ CA^{N-k+1}D & CA^{N-k+2}D & \cdots & CA^{N+1}D \\ CA^{N-k+2}D & CA^{N-k+3}D & \cdots & CA^{N+2}D \\ \vdots & \vdots & \ddots & \vdots \\ CA^{2N-k}D & CA^{2N-k+1}D & \cdots & CA^{2N}D \end{bmatrix}$$
$$= \begin{bmatrix} H_{N-k+1} & H_{N-k+2} & \cdots & H_{N+1} \\ H_{N-k+2} & H_{N-k+3} & \cdots & H_{N+2} \\ H_{N-k+3} & H_{N-k+4} & \cdots & H_{N+3} \\ \vdots & \vdots & \ddots & \vdots \\ H_{2N-k+1} & H_{2N-k+2} & \cdots & H_{2N+1} \end{bmatrix},$$

and $\mathbf{P}_{k+1}$ and $\mathbf{N}_{k+1}$ are as defined in (3.23) and (3.27). As can be seen, the amount of needed Markov parameters is increased from the first $N+1$ to the first $2N+1$. Every part is written in terms of the Markov parameters, except for the vector

$$\begin{bmatrix} CA^{N} \\ CA^{N+1} \\ CA^{N+2} \\ \vdots \\ CA^{2N} \end{bmatrix} \hat{x}_{0_i}.$$

If there is no a priori knowledge about the system, it is not possible to obtain this vector with a model based Kalman filter. However, there is a way to approximate this vector data-based with use of the estimated parameter P_1 from equation (4.13) as follows. The first element, $CA^N\hat{x}_{0_i}$ is already available since this is exactly the last element of the controller state vector $\overline{x}_{0_i}$. The remaining part of the matrix can be computed (using equations (4.5) and (4.6)) as

$$\overline{x}_{0_i} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} \hat{x}_{0_i} = \mathbf{O}_{N+1}\hat{x}_{0_i}, \qquad (5.10)$$
$$\hat{x}_{0_i} = \mathbf{O}_{N+1}^{\dagger}\overline{x}_{0_i}.$$

Now we can write

$$\begin{bmatrix} CA^{N+1} \\ CA^{N+2} \\ CA^{N+3} \\ \vdots \\ CA^{2N+1} \end{bmatrix} \hat{x}_{0_i} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} A^{N+1}\hat{x}_{0_i} = \mathbf{O}_{N+1}A^{N+1}\hat{x}_{0_i} \qquad (5.11)$$
$$= \mathbf{O}_{N+1}\left(-M\mathbf{O}_{N+1}\right)\mathbf{O}_{N+1}^{\dagger}\overline{x}_{0_i}.$$

The final equation can be approximated by assuming that $\mathbf{O}_{N+1}\mathbf{O}_{N+1}^{\dagger} = I_{N+1}$. Equation (5.11) can then be written as

$$\begin{bmatrix} CA^{N+1} \\ CA^{N+2} \\ CA^{N+3} \\ \vdots \\ CA^{2N+1} \end{bmatrix} \hat{x}_{0_i} = -\mathbf{O}_{N+1} M \overline{x}_{0_i} \tag{5.12}$$
$$= P_1 \overline{x}_{0_k}.$$

Every part is now expressed in data-based terms, this finishes the design of the moving horizon LQG controller since the feedback gains do not need any initialization and can be computed every horizon based on the estimated Markov parameters.

5.3 Simulations on a time-invariant system

In this section, some simulations will be done to illustrate the equivalence between the data-based controller and the model-based LQG controller. The simulation model is the same as in the previous chapter and can be found in appendix B. The observer is simulated in the next subsection.

5.3.1 Controller state observer

In this subsection, the controller state estimator is simulated. For this purpose, the model is subjected to an initial condition $x_0 = \begin{bmatrix} 1 & 2 & 0 & 0 \end{bmatrix}^T$. The Markov parameters (M and H) and the coefficient P_1 used for the data-based observer are computed with the use of the model to be able to make a comparison between the model-based and data-based control algorithm. The simulation scheme is depicted in figure 5.3.

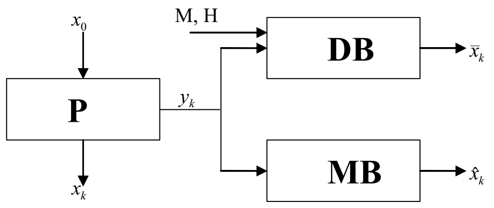


Figure 5.3: Simulation scheme, **P** denotes the model, **MB** denotes the model-based optimal observer and **DB** denotes the data-based observer.

The true and model-based estimated controller states are computed by pre-multiplying x_k and $\hat{x}_k$ by the matrix

$$\begin{bmatrix} C \\ CA \\ \vdots \\ CA^N \end{bmatrix},$$

while the data-based estimated controller states are computed using only the Markov parameters of the system. The tuning parameters are set to: $dt = 0.001$, $N = 5$, $W = 10^{10}$ and $V = 10^{-4}$. These extreme values are used to be able to illustrate the observing behavior in such a small time frame. In figure 5.4, the true, model-based and data-based estimated controllers states are depicted for two horizons, so for $k = 0, 1, 2, \ldots, 2N$.

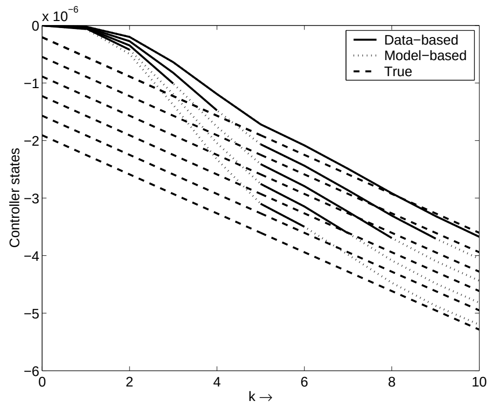


Figure 5.4: True, model-based and data-based controller state vector

As can be seen from the solid line in figure 5.4, the dimension of the data-based estimated controller vector is decreasing from $N+1$ to 2 in one horizon. After that, the vector is again initialized to $N+1$ values. The data-based and model-based controller vectors coincide and indeed go to the actual values.

5.3.2 Regulation

In this subsection, the regulator performance of the data- and model-based LQG controller are investigated. The simulation diagram is shown in figure 5.5. In addition to the initial condition of the system, x_0, it is also excited with white noise, n. The horizon is set to $N = 4$ and the weighting factors are set as follows: $W = 10^{10}$, $V = 10^{-4}$, $Q = 10^{4}$ and $R = 10^{-7}$. The sample time is $dt = 0.001$ and the Markov parameters are estimated from the input/output data. The estimated Markov parameters are filtered with a first order low-pass filter with cut-off frequency 100 Hz in order to obtain a smoother set of parameters.

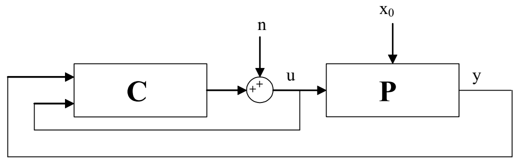
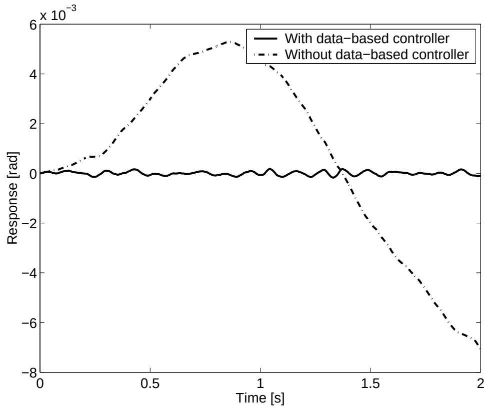
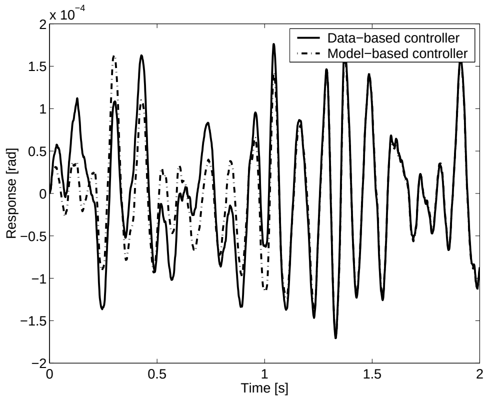


Figure 5.5: Simulation scheme for regulation, **P** denotes the model, **C** denotes the controller.

In figure 5.6(a), the response of the system with and without the data-based controller is depicted while in figure 5.6(b), the response of the system with the data- and model-based algorithm is showed.



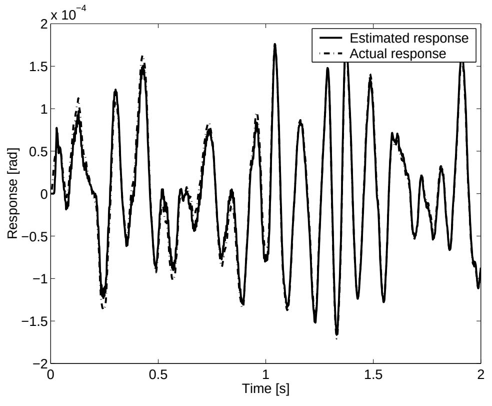
(a) Response with and without data-based controller



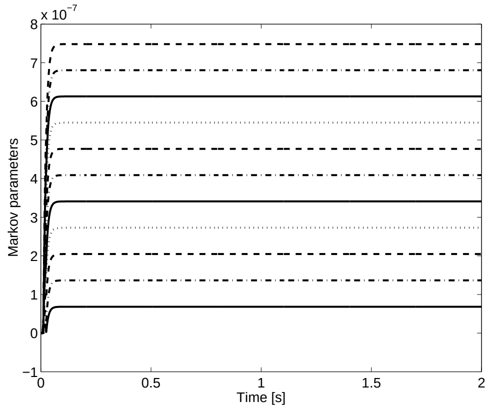
(b) Response with the model- and data-based controller

Figure 5.6: Regulator performance of the data-based controller

The regulator performs well, there is a small difference in the beginning between the data- and model-based controller since in this time, the Markov parameters and the coefficient P_1 used in the data-based controller are not converged yet. After they are converged, the data-based observer needs some time to converge to the actual controller states and when that is done, the output of the model- and data-based controllers coincide. In figure 5.7(a), the actual and estimated response (i.e., the first element of the controller state vector) is shown. The estimated Markov parameters are depicted in figure 5.7(b).



(a) Actual and estimated response



(b) Estimated Markov parameters

Figure 5.7: Response and Markov parameters of the system

5.4 Simulations on a time-varying system

In this section, some simulations will be done on the same system as used before, but now the gain of the system is slowly increasing. The simulation scheme can be found in figure 5.5. The bode diagrams of the original and final system are depicted in figure 5.8. In 2 seconds, the gain is gradually increased with 6 dB.

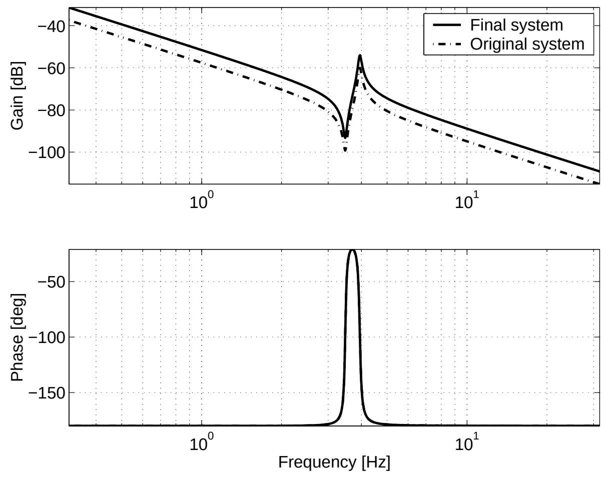


Figure 5.8: Original and final simulation model

The weighting of the states is increased to $Q = 10^6$ to obtain better regulation performance, if the same approach is used to initialize the data-based controller state vector as in the preceding subsection, so with the use of the coefficient P_1, the controller becomes unstable. This is caused by the fact that the estimation of the matrix P_1 from equation (5.12) does not produce accurate results and since this matrix is used to initialize the controller state every horizon, the system becomes unstable. There are several possible solutions to this problem. One can try to leave out the part

$$\begin{bmatrix} CA^N \\ CA^{N+1} \\ CA^{N+2} \\ \vdots \\ CA^{2N} \end{bmatrix} \hat{x}_{0_k},$$

from equation (5.9) but this can also lead to an unstable system when this part is too important to be left out. Another option is to initialize the controller state vector directly with an estimate based on the Markov parameter estimator. This approach, however, is not related with the LQG problem, it just provides another option to initialize the controller state vector every horizon and this can be done in the following way. As was stated in section 4.2, the current state can be expressed in terms of p past input and output values (see equation (4.7)),

$$\tilde{x}(k) = \sum_{i=1}^{p} \alpha_i u(k-i) + \beta_i y(k-i), \tag{5.13}$$

where $\tilde{x}(k)$ is the estimated state vector, $\begin{bmatrix} \alpha_p & \dots & \alpha_2 & \alpha_1 \end{bmatrix} = \mathbf{C}_p + \mathbf{M}\mathbf{T}_p$ and $\begin{bmatrix} \beta_p & \dots & \beta_2 & \beta_1 \end{bmatrix} = -\mathbf{M}$. Equation (5.13) is a state estimator and the coefficients α_i and β_i can be computed by means of the Hankel-Toeplitz model. In section 4.2 is explained, how the coefficients $P_1 = -\mathbf{O}_p\mathbf{M}$, $P_2 = \mathbf{O}_p(\mathbf{C}_p + \mathbf{M}\mathbf{T}_p)$ and a Hankel matrix $\mathbf{H} = \mathbf{O}_p\mathbf{C}_p$ can be obtained from input and output data. The observability matrix $\overline{\mathbf{O}}_p$ can be extracted from a singular value decomposition of $\mathbf{H} = U\Sigma V^T$,

$$\overline{\mathbf{O}}_p = U\Sigma^{\frac{1}{2}}. \tag{5.14}$$

Since the state-space representation is only equivalent up to a similarity transform of the state vector, $\overline{\mathbf{O}}_p$ is in a different set of coordinates than $\mathbf{O}_p$, but they are related by $\overline{\mathbf{O}}_p = \mathbf{O}_p T$, where

T is a transformation matrix. The identified parameter combinations, however, are invariant with respect to such a transformation since

$$\begin{aligned} P_1 &= -\mathbf{O}_p\mathbf{M} = -\mathbf{O}_pTT^{-1}\mathbf{M} = -\overline{\mathbf{O}}_p\overline{\mathbf{M}}, \\ P_2 &= \mathbf{O}_p\left(\mathbf{C}_p + \mathbf{M}\mathbf{T}_p\right) = \mathbf{O}_pT\left(T^{-1}\mathbf{C}_p + T^{-1}\mathbf{M}\mathbf{T}_p\right) = \overline{\mathbf{O}}_p\left(\overline{\mathbf{C}_p} + \overline{\mathbf{M}}\mathbf{T}_p\right). \end{aligned} \tag{5.15}$$

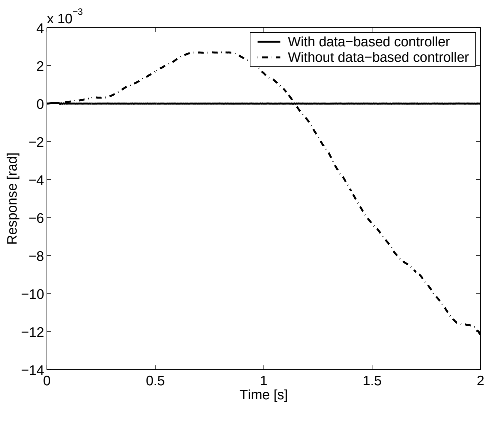
The needed parameters to construct a state estimator can be found from

$$\overline{\mathbf{O}}_p \left[\begin{array}{cc} \left(\overline{\mathbf{C}}_p + \overline{\mathbf{M}}\mathbf{T}_p\right) & -\overline{\mathbf{M}} \end{array}\right] = \left[\begin{array}{cc} P_2 & P_1 \end{array}\right]. \tag{5.16}$$

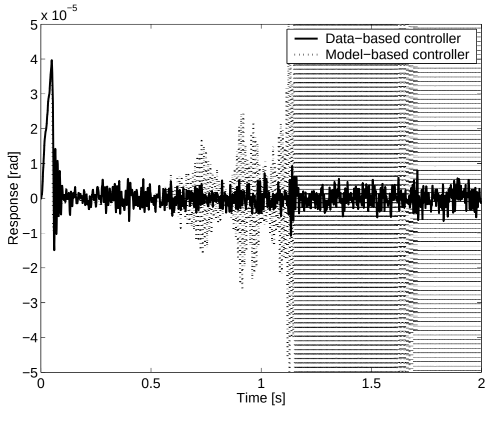
This data-based state estimator will generally not produce state estimates according to the coordinate representation of the LQG controller but the controller state vector is invariant to the similarity transformation since

$$\mathbf{O}_p\hat{x}_k = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^N \end{bmatrix}\hat{x}_k = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^N \end{bmatrix}TT^{-1}\hat{x}_k = \overline{\mathbf{O}}_p\tilde{x}_k, \tag{5.17}$$

where $\overline{\mathbf{O}}_p$ and $\tilde{x}_k$ are given by equations (5.14) and (5.13). This vector is now used to initialize the controller state vector each horizon. Therefore, the value for p should be set to $p = N + 1$ and the value for L is then chosen to be the minimal $L = 3p$ since there is not much noise present in the data and the computational effort increases when L is increased. Since it takes some time to fill all matrices with data to estimate all parameters, the controller is switched on after 0.05 seconds. The unregulated and data-based regulated response of the system is shown in figure 5.9 (a).



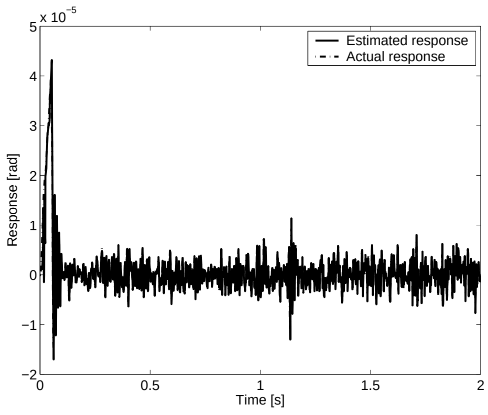
(a) Response with and without data-based controller



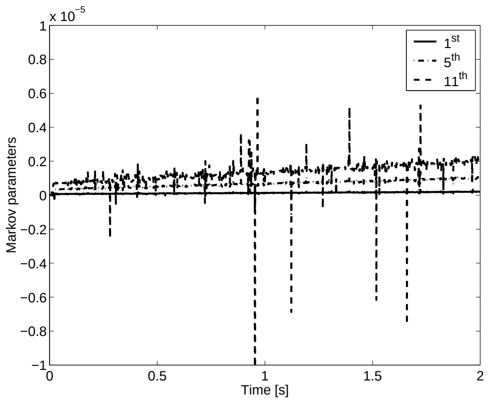
(b) Response with the model- and data-based controller

Figure 5.9: Regulator performance

In figure 5.9 (b), both responses of the data-based and model-based controlled systems are depicted. As can be seen, the model-based controlled system becomes unstable while the data-based controlled system behaves fine. This is due to the fact that the data-based LQG controller adjusts itself to the actual dynamics, while the model-based controller does not. In figure 5.10(a) and (b) the actual and the estimated response and the 1^{st}, 5^{th} and the 11^{th} estimated Markov parameters are depicted.



(a) Actual and estimated response



(b) Estimated Markov parameters

Figure 5.10: Regulator performance of the data-based controller

Since both lines coincide in the first figure, it is obvious that the response (i.e. the first element of the controller state vector) is estimated quite well with the use of the new approach. The estimated Markov parameters indeed show that the gain of the system is increasing. They are a bit noisy, since the data from which the Markov parameters are estimated never contains an actual linear system since this system is varying constantly and therefore the data is not representable for a single linear system.
A great advantage of the data-based controller compared to the model-based LQG controller is that the first one is able to adjust itself every horizon to the actual dynamics while the second one is a static controller which can become unstable when the dynamics are changing. Of course, this not only holds for changes in gain but also in resonance frequencies, damping, etc.

5.5 The data-based LQG regulator as a tracking controller

Instead of taking the output of the system as input to the controller, the error can be chosen as the input to the controller. This way, it should be possible not only to regulate the output of a system to zero, but to regulate the error to zero as well. Therefore, the input to the data-based controller can be chosen to be the input to the system and the measured error. The controller then estimates the Markov parameters from the input to the error and generates control action to regulate this error to zero. The scheme is depicted in figure 5.11

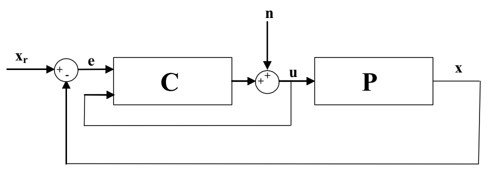
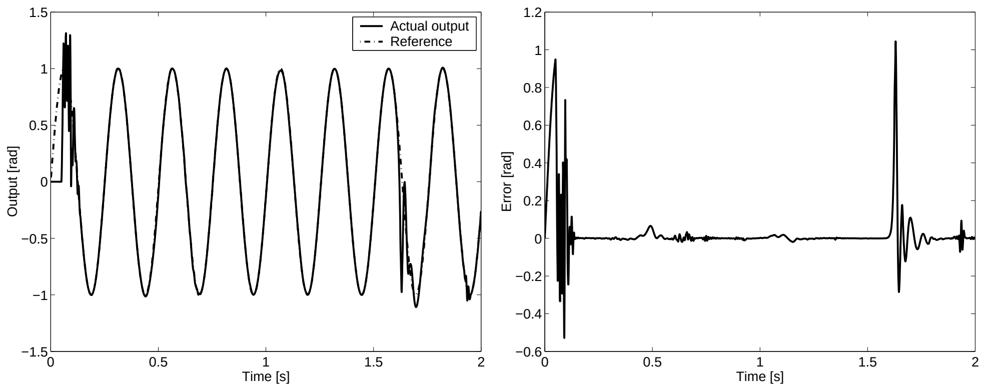


Figure 5.11: Scheme to use the data-based regulator for tracking

It should be noted that this approach is only feasible if the reference is varying slowly with respect to the length of the data which is used to estimate the Markov parameters since the reference acts as a disturbance on this data. If the reference is changing rapidly the Markov estimation process tries to explain this behavior of the data, based on the ARMarkov models in which the reference is not taken into account. A solution can be to extend the ARMarkov models

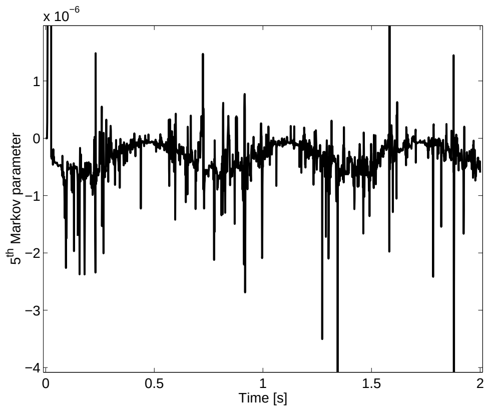
in such a way that the reference is not taken into account while estimating the parameters, but is taken into account when estimating the controller state.
White noise should be added to guarantee that the input is sufficiently exciting to estimate the needed parameters. The gain of the example system is now varied in a sinusoidal way, it is increased and decreased between $\frac{1}{5}\times$ and $5\times$ the nominal value. The tuning parameters are set to: $N = 4$, $W = 10^{10}$, $V = 10^{-4}$, $Q = 10^{6}$ and $R = 10^{-7}$. The sample time is $dt = 0.001$ and the controller is switched on after 0.05 seconds to make sure the Markov parameter estimator is fully initialized. In figure 5.12(a), the reference and the output of the system are depicted, while in figure 5.12(b) the error is depicted.



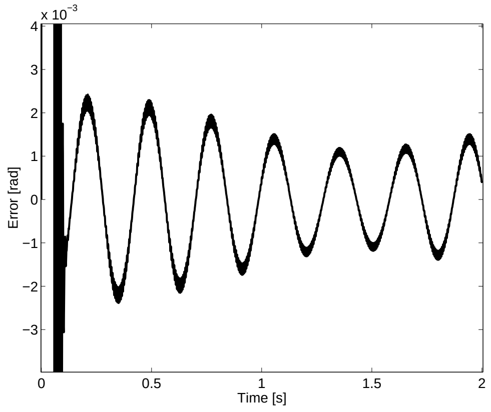
(a) Reference and actual output. (b) Tracking error

Figure 5.12: Tracking performance of the data-based controller

The poor tracking at some moments is due a combination of factors, such as the fact the gain of the system at some moments is very low which this results in lower gains for the controller state estimator and for the feedback (since the input weighting R becomes relatively more important) and the fact that the horizon is very small. In figure 5.13(a), the 5^{th} estimated Markov parameter is shown. The sinusoidal gain change is visible, the noise is probably introduced by the changing dynamics of the system which result in non-consistent data for a single linear system, used for the estimation process. For time-invariant systems, the tracking performance is very nice as can be seen in figure 5.13(b), where the error is depicted when the system is not varying. The error is changing in a sinusoidal way, probably caused by the influence of the two sinusoidal functions: the reference and the multiplication factor for the gain.



(a) Estimated 5^{th} Markov parameter.



(b) Tracking error when the system is time-invariant.

Figure 5.13: Tracking performance of the data-based controller

It is obvious that the data-based controller can be used as a tracking controller, if the remarks in the beginning of this section are kept in mind. In the next chapter, the controller wil be tested on a real setup, namely the RRR-robot in the DCT-lab.

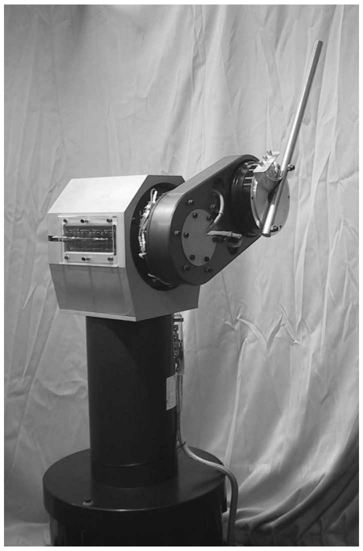
Chapter 6

The RRR-robotic arm: Simulations and experiments

In this chapter, simulations and experiments on the experimental setup are performed. The setup is an RRR-robotic arm and is presented in the next section together with three possible schemes to control it. After that, simulations and experiments are carried out for each of these schemes.

6.1 The experimental setup

The RRR-robotic arm, installed in the laboratory of the section Dynamics and Control Technology of the Department of Mechanical Engineering at the Technisch Universiteit Eindhoven, is a robotic arm with three rotational joints, implemented as a waist, shoulder, and elbow. The robot is equipped with three brushless DC direct drive motors and slip rings to obtain unconstrained motion. The amplifiers and encoders are connected to a PC-based rapid control prototyping system with which controllers can be implemented that are designed in Matlab/Simulink. The RRR-robot and its kinematic representation are depicted in figure 6.1.



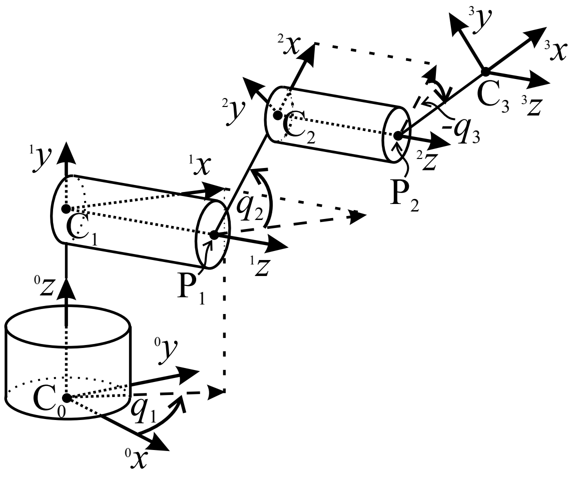


Figure 6.1: Picture and kinematic representation of the RRR-robot

A more detailed description of the robotic arm is given in [2] and [3]. For the forward and inverse robot kinematics models, see [5]. The rigid body dynamics of the robot can be represented using a standard Euler-Lagrange formalism [14] by

$$\boldsymbol{\tau}(t) = \mathbf{M}(\mathbf{q}(t))\ddot{\mathbf{q}}(t) + \mathbf{C}\left(\mathbf{q}(t), \dot{\mathbf{q}}(t)\right)\dot{\mathbf{q}}(t) + \mathbf{g}(\mathbf{q}(t)) + \boldsymbol{\tau_f}\left(\mathbf{q}(t), \dot{\mathbf{q}}(t)\right) \tag{6.1}$$

where $\boldsymbol{\tau}$ is a 3×1 vector of control torques, $\mathbf{M}$ is a 3×3 inertia matrix, $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ are 3×1 vectors of joint motions, velocities, and accelerations, respectively, and $\mathbf{C}\dot{\mathbf{q}}$, $\mathbf{g}$, and $\boldsymbol{\tau_f}$ are the 3×1 vectors of Coriolis/centripetal, gravitational, and friction forces. The robot is a highly coupled system, for most configurations the mass matrix is non-diagonal which implies that a torque applied by one motor will result in movement of all three joints.

6.1.1 Control schemes

Basically three configurations for the control of the RRR robot are used, namely decentralized control, control using model-based feedforward and control with the use of feedback linearization. In this section, these three setups will be presented. In these schemes, the controller is denoted by $\mathbf{C}$ and the system to be controlled by $\mathbf{P}$. To avoid unnecessary complexity in the equations and without loss of generality, the time argument (t) will be omitted from now on.

Decentralized control

In the case the robot is controlled by decentralized control, the controller acts directly on the system without any additional feedforward or feedback compensators present. The control law in this case is

$$u = u_c, \tag{6.2}$$

where u_c is the controller output and is still undefined. A scheme is depicted in figure 6.2.

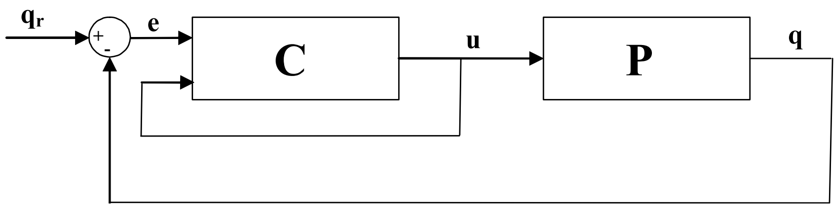


Figure 6.2: Decentralized control scheme.

Model-based feedforward

Another way to track a desired trajectory $\mathbf{q_r}$, is to use a model-based feedforward that predicts the required torque in combination with a feedback controller which stabilizes the robot and rejects the effect of model errors and disturbances. This can be done by applying the following control law, where u_c still is to be defined,

$$u = \mathbf{M}(\mathbf{q}_r)\ddot{\mathbf{q}}_r + \mathbf{C}\left(\mathbf{q}_r, \dot{\mathbf{q}}_r\right)\dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}_r) + \boldsymbol{\tau_f}\left(\mathbf{q}_r, \dot{\mathbf{q}}_r\right) + u_c \tag{6.3}$$

which leads to the following closed loop equation

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}\left(\mathbf{q}, \dot{\mathbf{q}}\right)\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau_f}\left(\mathbf{q}, \dot{\mathbf{q}}\right) = \mathbf{M}(\mathbf{q}_r)\ddot{\mathbf{q}}_r + \mathbf{C}\left(\mathbf{q}_r, \dot{\mathbf{q}}_r\right)\dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}_r) + \boldsymbol{\tau_f}\left(\mathbf{q}_r, \dot{\mathbf{q}}_r\right) + u_c \tag{6.4}$$

The scheme is depicted in figure 6.3.

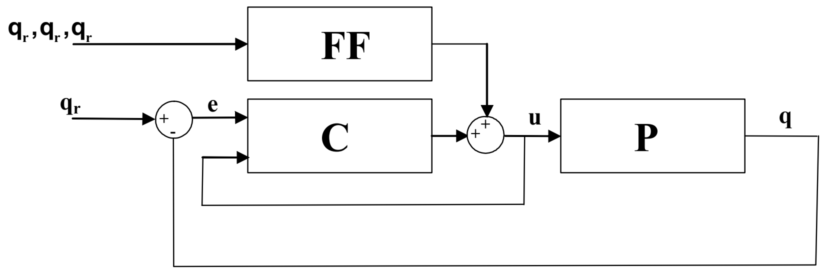


Figure 6.3: Model-based feedforward control scheme.

Feedback linearization

Feedback linearization is a method that creates an inner loop based on the robot model, which decouples and linearizes the robot. An outer loop controller should be designed around this inner loop which achieves the desired closed loop behavior. It is also called inverse dynamics control. The control law used for feedback linearization is

$$u = \mathbf{M}(\mathbf{q})\mathbf{v} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau_f}(\mathbf{q}, \dot{\mathbf{q}}) \tag{6.5}$$

where $\mathbf{v}$ is still to be defined, the closed loop dynamics become

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})\mathbf{v} \tag{6.6}$$

and since $\mathbf{M}(\mathbf{q})$ is a regular matrix for all $\mathbf{q}$, this is equivalent to

$$\ddot{\mathbf{q}} = \mathbf{v} \tag{6.7}$$

This represents a set of three SISO systems, each behaving as a double integrator. Around this inner loop, a control law v must be designed which achieves the desired closed loop behavior. In the presence of measurement noise or delays, another possibility is to use the reference position and velocity instead of the measured ones to compensate the nonlinearities. Now the control law becomes

$$u = \mathbf{M}(\mathbf{q}_r)\mathbf{v} + \mathbf{C}(\mathbf{q}_r, \dot{\mathbf{q}}_r)\,\dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}_r) + \boldsymbol{\tau_f}(\mathbf{q}_r, \dot{\mathbf{q}}_r) \tag{6.8}$$

A simple and effective outer loop control law is a combination of acceleration feedforward and the controller output

$$\mathbf{v} = \ddot{\mathbf{q}}_r + u_c \tag{6.9}$$

The difference with the model-based feedforward control scheme is that the feedback controller is multiplied with the position dependent non-diagonal mass matrix. The closed loop dynamics with feedback linearization are

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau_f}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{M}(\mathbf{q}_r)\mathbf{v} + \mathbf{C}(\mathbf{q}_r, \dot{\mathbf{q}}_r)\,\dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}_r) + \boldsymbol{\tau_f}(\mathbf{q}_r, \dot{\mathbf{q}}_r) \tag{6.10}$$

The feedback linearized control scheme is shown in figure 6.4.

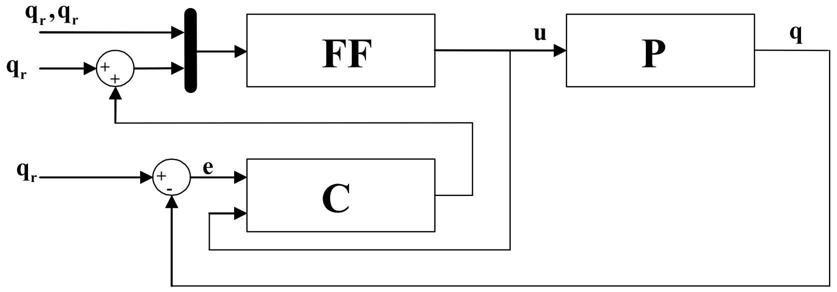


Figure 6.4: Feedback linearized control scheme

6.2 Simulations with the RRR-robotic arm model

In this section, simulations are done on the RRR-robotic arm. The forward and inverse kinematic models of the RRR are made by Dragan Kostić and are used here to simulate the system. Furthermore, all simulations and experiments are carried out on the first joint of the RRR, so q_1 is controlled (see figure 6.1). The other two joints are controlled by a PD controller and are not discussed any further. The references are shown in figure 6.5, they have cosine velocity profiles with zero initial and terminal speeds and accelerations.

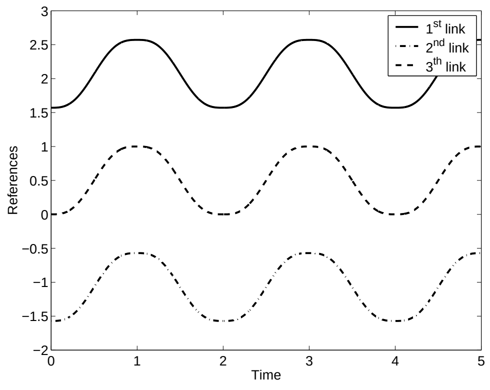
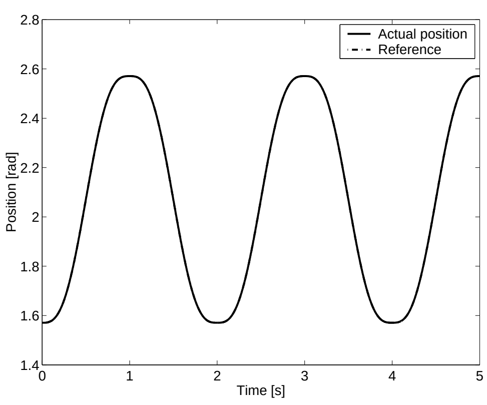


Figure 6.5: Reference trajectories

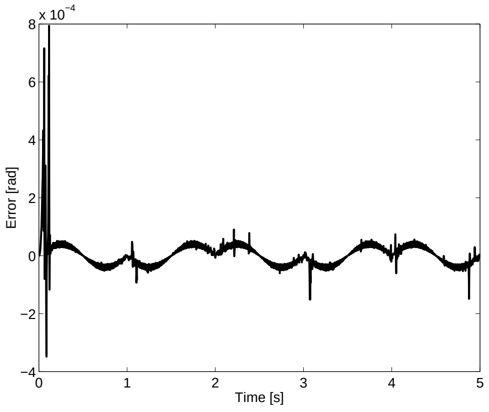
In all simulations, the sample time is $dt = 0.001$ second and the horizon is set to $N = 5$ samples. It would be better to use a larger horizon but this cannot be done because of the computational effort which is involved then. White noise is added to the control output to make sure the data is sufficiently rich for the estimation of the parameters. Since the control horizon is small and the controller state vector is initialized every horizon by equation (5.17), the observer weighting matrices W and V are kept constant at $W = 10^3$ and $V = 10^{-6}$. In the following subsections the three control schemes are presented, starting with the feedback linearization scheme since this scheme requires the least challenge for the controller.

6.2.1 Feedback linearized control

Since feedback linearization decouples the system, it requires the least challenge to control the system. The system to be controlled is now described by a double integrator. The parameters are: $Q = 10^6$, $R = 10^{-6}$, $W = 10^3$ and $V = 10^{-6}$. In figure 6.6(a) the reference and actual position for the first joint are depicted, figure 6.6(b) depicts the tracking error. As can be seen, the data-based controller shows a good tracking behavior.



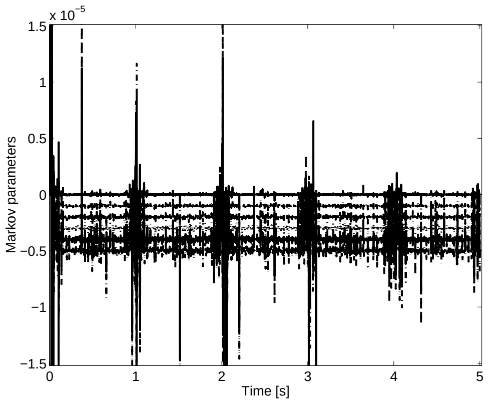
(a) Reference and actual position.



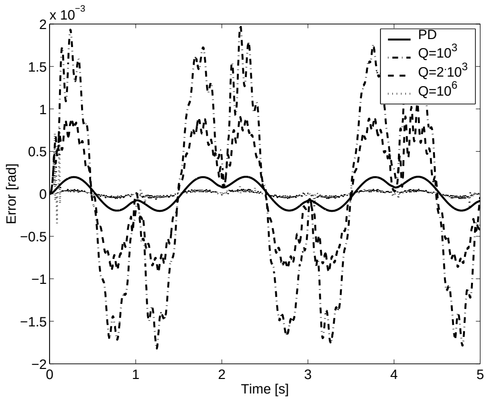
(b) Tracking error.

Figure 6.6: Tracking simulation of the RRR model with feedback linearization.

The first six estimated Markov parameters are depicted in figure 6.7(a). They indicate a linear time-invariant system since the parameters appear to be constant. To see the effect of different weightings and to compare the result to a PD controller, figure 6.7(b) shows the tracking error for various values of the weighting matrix Q and a PD controller ($K_p = 1000$, $K_d = 2\sqrt{K_p}$).



(a) First six estimated Markov parameters.



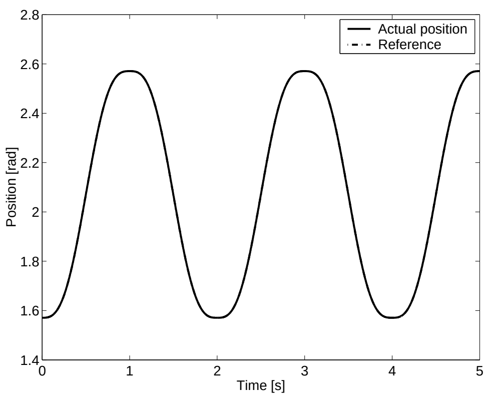
(b) Tracking error with various values for Q.

Figure 6.7: Markov parameters and tracking errors for various values of Q.

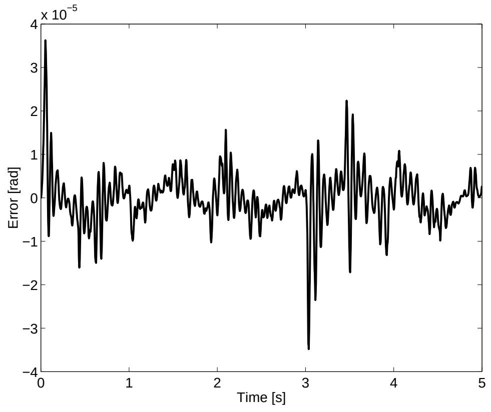
The error is expected to converge to zero since the used forward and inverse kinematics are exactly each other inverse. The influence of different weightings is obvious in the tracking error, the more weight is put on the error, the better the tracking performance is. In the following subsection, the tracking on the scheme with model-based feedforward is presented.

6.2.2 Control with model-based feedforward

In this subsection, similar simulations are performed, but now with the use of model-based feedforward instead of feedback linearization, as presented in section 6.1.1. The parameters are: $Q = 10^3$, $R = 10^{-6}$, $W = 10^3$ and $V = 10^{-6}$. In figure 6.8(a) the reference and actual position for the first joint are depicted. Figure 6.8(b) depicts the tracking error. As can be seen, the data-based controller performs well.



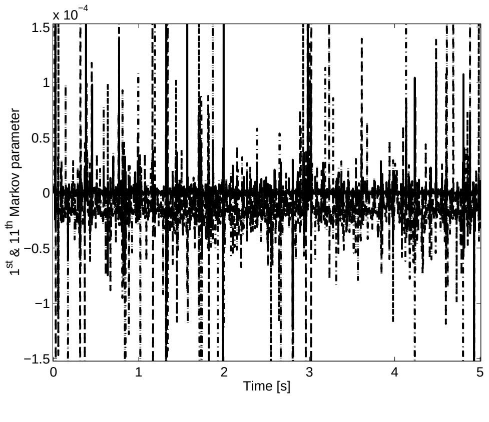
(a) Reference and actual position.



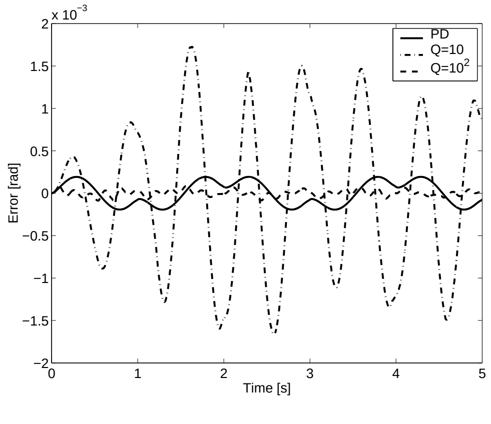
(b) Tracking error.

Figure 6.8: Tracking simulation of the RRR model with the use of model-based feedforward.

The first and the eleventh estimated Markov parameter are depicted in figure 6.9(a) and they are much more noisy then was the case with feedback linearization. This is due to the fact the control action is not multiplied with the mass matrix and thus that the influence of the second and third joint is present in the measured data from the first, data-based controlled, joint. To see the effect of different weightings and to compare the result to a PD controller, figure 6.9(b) shows the tracking error for various values of the weighting matrix Q and a PD controller ($K_p = 1000$, $K_d = 2\sqrt{K_p}$).



(a) First and eleventh Markov parameter.



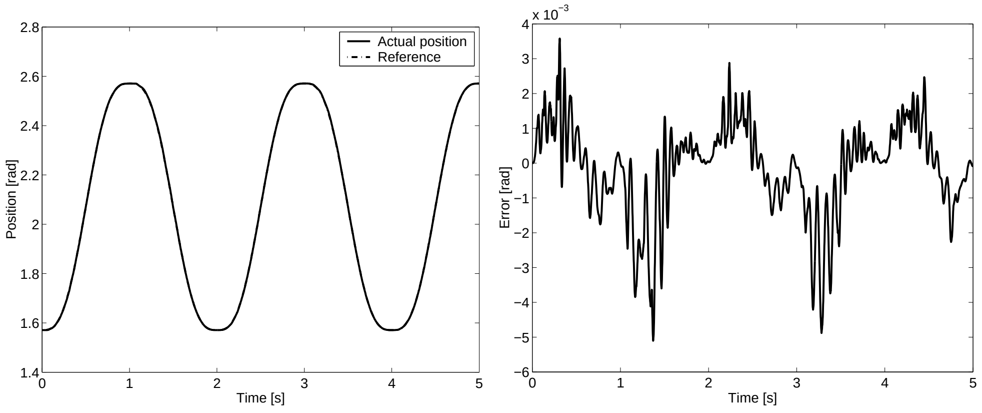
(b) Tracking error with various values for Q.

Figure 6.9: Markov parameters and tracking errors for various values of Q

The influence of different weightings is obvious in the tracking error, the more weight is put on the error, the better the tracking performance is. The differences in the values for Q with respect to those in case of feedback linearization, are due to the different gains of both control schemes, since the controller is multiplied with the mass matrix when feedback linearization is present. In the following subsection, the tracking on the decentralized control scheme is presented.

6.2.3 Decentralized control

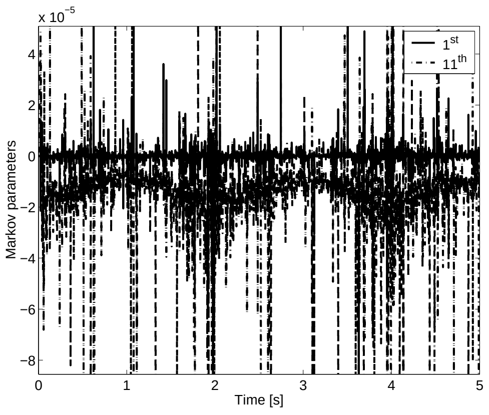
In this subsection, decentralized control simulations are presented. The parameters are: $Q = 10^3$, $R = 10^{-6}$, $W = 10^3$ and $V = 10^{-6}$. In figure 6.10(a) the reference and actual position for the first joint are depicted. Figure 6.10(b) depicts the tracking error. As can be seen, the data-based controller performs well.



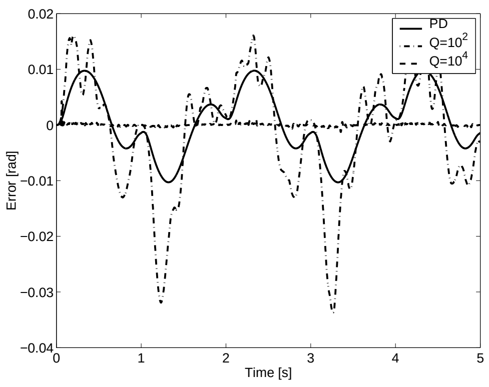
(a) Reference and actual position. (b) Tracking error.

Figure 6.10: Tracking simulation of the RRR model with the use of decentralized control.

The first and the eleventh estimated Markov parameter are depicted in figure 6.11(a). They are noisy since the system under consideration is non-linear and therefore the Markov parameter estimation algorithm (which makes use of linear models) is not able to estimate a representable set of parameters. There is some sinusoidal trend visible in the Markov parameters. This is probably due to the fact that the reference is a sinusoid and therefore, the other joints influence the data in the first joint in a sinusoidal way. To see the effect of different weightings and to compare the result to a PD controller, figure 6.11(b) shows the tracking error for various values of the weighting matrix Q and a PD controller ($K_p = 1000$, $K_d = 2\sqrt{K_p}$).



(a) First and eleventh Markov parameter.



(b) Tracking error with various values for Q.

Figure 6.11: Markov parameters and tracking errors for various values of Q

The influence of different weightings is obvious in the tracking error, the more weight is put on the error, the better the tracking performance is. The simulations show promising results, the next step will be to perform real-time experiments on the RRR robotic arm. These experiments are presented in the next section.

6.3 Experiments on the RRR-robotic arm

In this section, experiments are done on the RRR-robotic arm. All experiments are carried out on the first joint of the RRR, so q_1 is controlled (see figure 6.1). The three control schemes, presented in section 6.1.1, are also used during the experiments. However, the second and third joint are not controlled since it is observed that controlling them with PD controllers, if there is no feedback linearization present, results in large disturbances with high frequencies on the first joint. Because of these large high frequent disturbances in the data, the data-based controller is not able to correctly estimate the Markov parameters and the system becomes unstable. Since there are large differences between the performance during simulations and during experiments, no comparison is made. The reference signal and all controller settings differ from those during the simulations . Only the first joint is controlled with a reference signal represented by a sinusoid, shown in figure 6.12.

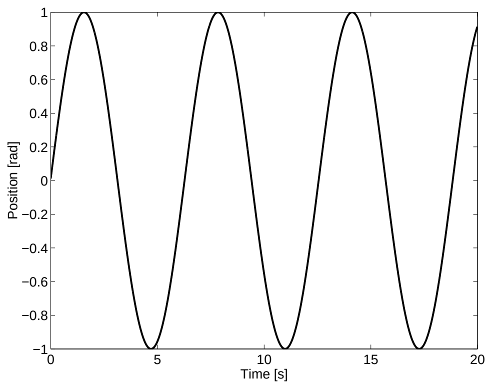
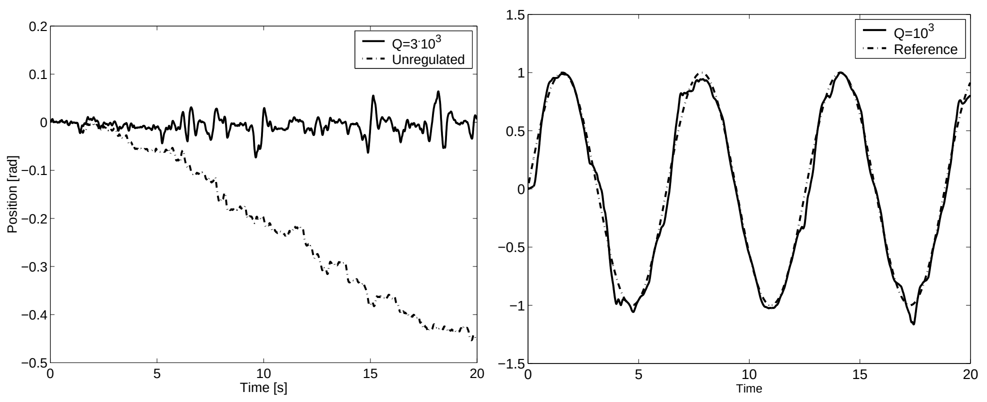


Figure 6.12: Reference trajectory

In all experiments, the sample time is $dt = 0.006$ second and the horizon is set to $N = 4$ samples. Because of the computational burden, it is not possible to sample with a higher frequency or to increase the horizon. White noise is added to the control output to make sure the data is sufficiently rich for the estimation of the parameters. The following subsections deal with the three control schemes, starting with the feedback linearization scheme. In each case, a regulation experiment to control the output to zero and a tracking experiment will be presented.

6.3.1 Feedback linearized control

In this subsection, the experiments are carried out with the presence of feedback linearization. Since a real system is controlled now and a model cannot perfectly describe it, the resulting dynamics will not be a double integrator but something like it with some additional resonances. Only the results of the data-based LQG controller are presented, so no comparison is made with the PD controllers. First, the regulation experiment is carried out. The parameters are determined by trial-and-error and the values are: $Q = 10^3$, $R = 10^{-3}$, $W = 10^4$and $V = 10^{-3}$. The output from the uncontrolled and controlled system over the first 20 seconds is shown in figure 6.13(a).

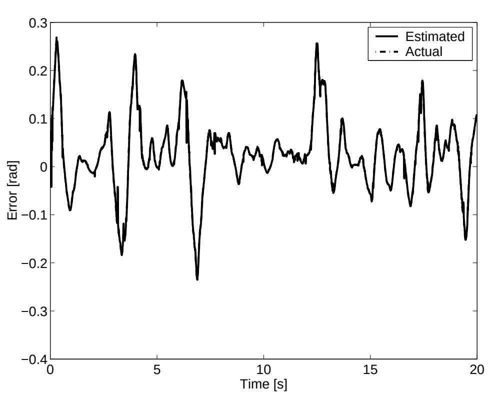


(a) Regulated and unregulated output.

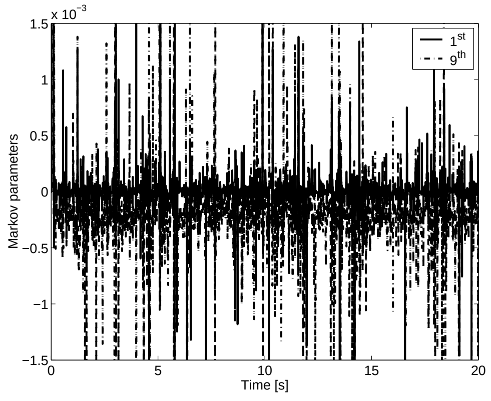
(b) Reference and output of the controlled system.

Figure 6.13: Regulation and tracking performance on the RRR.

Figure 6.13(b) shows the results of a tracking experiment. The data-based controller actually stabilizes the system. The tracking is not perfect but that can be expected from experiments which are performed with such a small horizon and a low sampling frequency. The actual and observed tracking errors are depicted in figure 6.14(a). As can be seen, the data-based observer performs very well. Model-based observers are expected to have smooth estimates of the states since the data is filtered by this model, however, when the data-based observer is used, the Markov parameters (and thus the representation of the system) is changing constantly and therefore no smooth estimation is obtained. The first and ninth estimated Markov parameters are shown in figure 6.14(b). It is not visible very well in the figure, but they seem to be constant, indicating a more or less time-invariant system.



(a) Actual and observed tracking error.

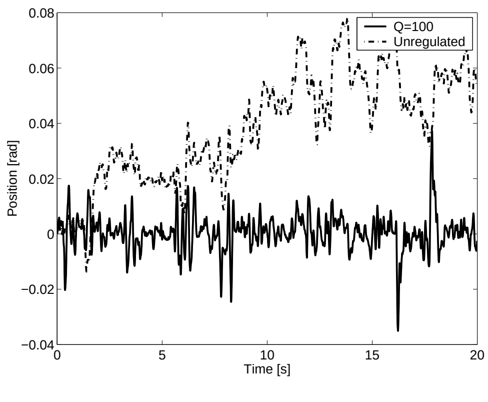


(b) The first and ninth estimated Markov parameter.

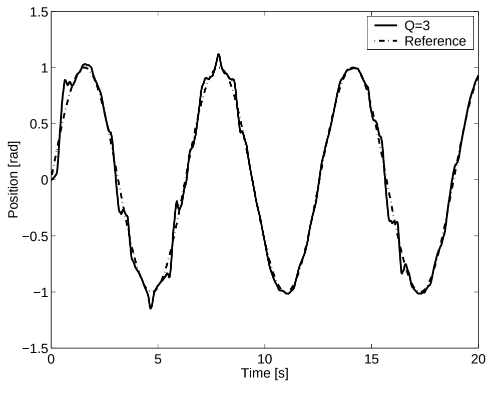
Figure 6.14: Performance of the observer and two estimated Markov parameters.

6.3.2 Control with model-based feedforward

In this experiment, model-based feedforward is used in addition to the data-based controller. The parameters are again determined by trial-and-error and set to $R = 10^{-3}$, $W = 10^4$ and $V = 10^{-3}$. The value of the weighting matrix Q depends on the experiment. During regulation it is set to $Q = 100$. The result is depicted in figure 6.15(a). Obviously, this control scheme is able to regulate the systems output. The value for Q during the tracking experiment is set to $Q = 3$ and this result is depicted in figure 6.15(b).



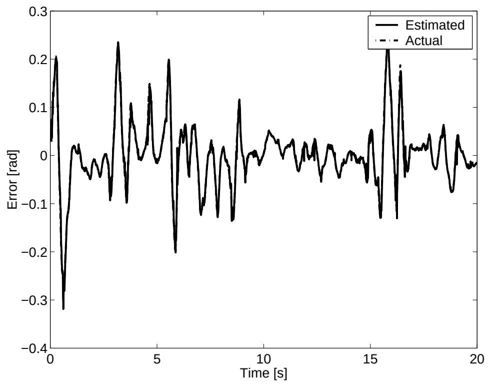
(a) Regulated and unregulated output.



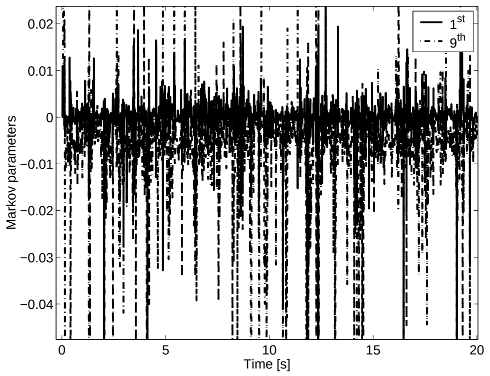
(b) Reference and output of the controlled system.

Figure 6.15: Regulation and tracking performance on the RRR.

As can be seen, the data-based controller is able to perform some kind of tracking. The actual and estimated tracking error are depicted in figure 6.16(a) while the first and last estimated Markov parameter are shown in figure 6.16(b). The observer is able to estimate the error correctly. The Markov parameters show less constant behavior than was the case with feedback linearization.



(a) Actual and observed tracking error.



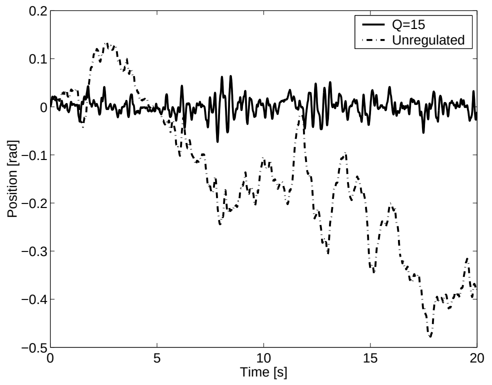
(b) The first and last estimated Markov parameters.

Figure 6.16: Performance of the observer and two estimated Markov parameters.

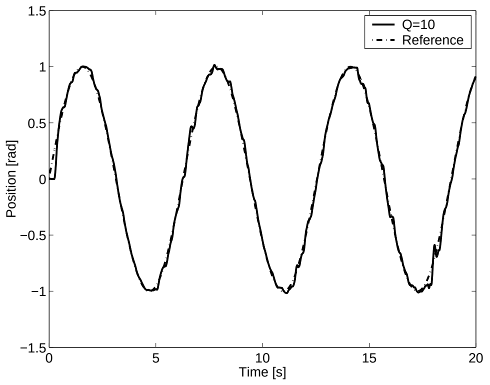
The real challenge is to control systems without any knowledge or use of a model. This is done in the next subsection, where decentralized control is used to control the RRR robotic arm.

6.3.3 Decentralized control

In this experiment, no a-priori knowledge of the system is used. The parameters are again determined by trial-and-error and set to $R = 10^{-3}$, $W = 10^4$ and $V = 10^{-3}$. The value of the weighting matrix Q depends on the experiment. During regulation it is set to $Q = 15$, while during tracking, $Q = 10$. The results of both experiments are depicted in figures 6.17(a) and (b).



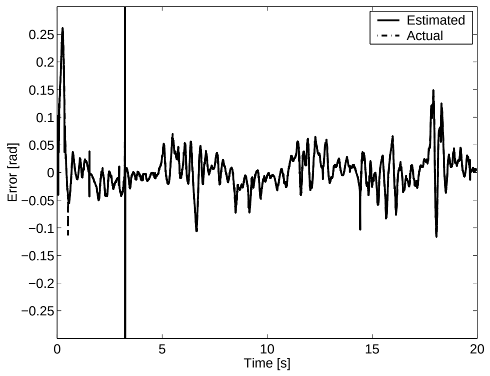
(a) Regulated and unregulated output.



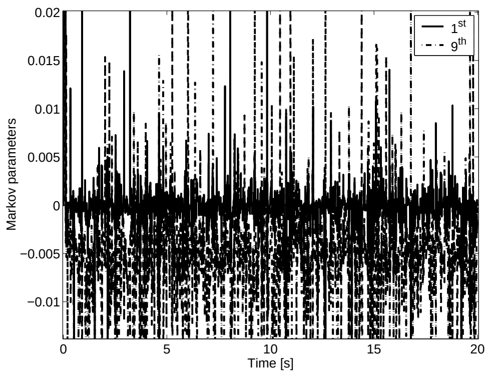
(b) Reference and output of the controlled system.

Figure 6.17: Regulation and tracking performance on the RRR.

It is obvious from both figures that the controller is able to regulate the systems output and to perform tracking, while absolutely no information of the system, except for the measured signals, is used to generate the controller. The actual and estimated tracking error are depicted in figure 6.18(a), with this setup, the observer is also able to estimate the error quite nicely. The first and last Markov parameters are shown in figure 6.18(b).



(a) Actual and observed tracking error.



(b) The first and ninth estimated Markov parameters.

Figure 6.18: Performance of the observer and the first and last estimated Markov parameters.

If more weight is put on the tracking error, the control action will become more aggressive. An experiment is carried out where Q is set to $Q = 100$. The reference and actual output are depicted in figure 6.19.

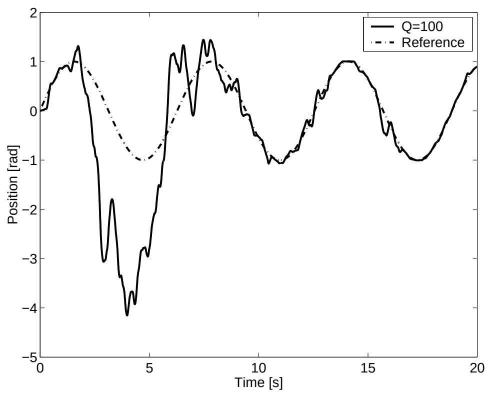


Figure 6.19: Aggressive control, $Q = 100$

The controller setting results in an aggressive manner on error reduction. This induces large gains and the overall result is worse, however, the system remains stable and tracking is performed. From all simulations and experiments, it can be concluded that the data-based LQG controller works. The observer is able to observe the controller state and the controller is able to reduce these. It should be possible to control any given system with this controller by just adding it and choosing some values for the weighting matrices.

Chapter 7

Conclusions and Recommendations

In this project, a data-based LQG controller has been considered and implemented in the Matlab/Simulink environment to perform simulations and experiments. The following conclusions and recommendations for future research can be made.

7.1 Conclusions

All the objectives stated in the problem formulations are carried out and from these, the following conclusions can be made. The model-based LQG algorithm is rewritten in terms of Markov parameters using closed form solutions to the Ricatti equations and by the introduction of a controller state vector. This resulted in the design of a data-based discrete time finite horizon linear quadratic gaussian controller. The results of the model- and data-based LQG controller are exactly the same when accurate Markov parameters are used.

An algorithm to estimate these parameters is presented in chapter four. It is based on the representation of the input/output data using ARMarkov models. Simulations show that the estimated Markov parameters coincide with the computed ones. Some time is needed before the parameters are converged and the implemented algorithm is computational demanding which limits the amount of parameters that can be estimated.

The estimated Markov parameters are used to compute the discrete finite time data-based controller. However, to control over infinite time, a moving horizon controller is developed where the estimated controller state vector is initialized every horizon with its last value. This is done using another estimated parameter, obtained with the algorithm to estimate the Markov parameters. The optimal gains are computed every new horizon based on the latest estimated Markov parameters. This results in a time varying data-based controller which adapts itself to the actual dynamics of the system under consideration. Since for time-varying systems, the initialization in this way is not possible due to the variance in the estimation of the used parameter, the controller state is initialized using a direct estimate of this state, based on the Markov parameter estimation algorithm.

Simulations show that in case of time-invariant systems, the model- and data-based optimal control inputs are equivalent. When the system is time-varying, the model-based controller can become unstable while the data-based controller adapts to the changing dynamics and remains stable. This means that the data-based controller can be used to control time-varying and even non-linear systems since it computes the control action based on the actual measured data, in which all information about the dynamics is available.

Furthermore, model-based LQG controllers are not able to perform tracking on a system, without extra precautions. Since the input and output of the data-based controller can be taken from any place in the control scheme, it is possible to obtain a tracking controller which regulates the error to zero. At the moment, this only works when the reference is slowly varying with respect to the length of the data-set used to estimate the Markov parameters and the controller state vector used to initialize the observer. This is due to the fact that the reference acts as a disturbance signal to the data used for this estimation.

Finally, some experiments have been done on the RRR robotic arm and the data-based controller is able to perform regulation and tracking of the first joint of this system. The tracking is worse than obtained with a nicely tuned PD controller. Since the controller is truly data-based, it is possible to use the controller for any given system without the necessity of a valid model of this system. This the great advantage of the data-based controller.

7.2 Recommendations

The gains used to compute the state feedback and observer are the optimal ones. However, in the case of small horizons, sub-optimal control is preferred since then the gains are not in their transient fase anymore but their steady-state optimal solution is used. In model-based LQG control, these steady state values can be computed using the algebraic Ricatti equations. The data-based controller needs closed-form solutions to the Ricatti equations. More research should be done in this field to determine the steady-state optimal gains using these closed-form solutions without the need for a large horizon.

At this moment, the used implementation of the algorithm to estimate the Markov parameters and the controller state vector is not very efficient. Quite a duplication of the data is needed and operations such as the pseudo-inverse and singular value decomposition are used to solve systems of linear equations, while other, more efficient, routines (such as rank one updates) are available. Implementation of these routines and the use of less data to obtain the same results should result in a less computational involving algorithm.

In order to obtain better results controlling the system, a larger horizon should be taken into account. Another possible solution could be the use of subsampling. These partly solve the problem with the low gains in the transient phase and this way also more frequencies can be taken into account by the controller. A higher sampling frequency is advised as well. These suggestions can only be implemented when the algorithm is made less computationally expensive or when the system used to implement the controller (i.e. computer) operates faster.

If the data-based LQG controller is used as a tracking controller, the algorithm to estimate the Markov parameters should be extended in such a way that reference signals are taken into account. This way, the reference signal does not disturb the estimation process anymore and therefore, it is no longer necessary that the reference signal is slowly varying with respect to the length of the control horizon.

More research can be done regarding pre-filtering the data or estimated parameters to obtain smoother results. For example, an exponential forgetting factor can be implemented in the estimation process.
At the moment, stability remains an open issue with respect to the data-based LQG controller. Since the control action is based upon measured data, large disturbances on the system or in the measured signals can lead to destabilization of the closed-loop system. This subject should be further investigated in order to guarantee stability.

The final recommendation is to implement other controllers with the use of input/output data.

More research should be done to determine if a model predictive controller, based on Markov parameters, or the design of H_∞-controllers can be done using input/output data.

Appendix A

Matrix Algebra

A.1 Schur forms

Consider a partition symmetric matrix

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix}. \tag{A.1}$$

Then the Schur decompositions are defined as

$$\begin{aligned} A_1 &= \begin{bmatrix} I & A_{12}A_{22}^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} A_{11} - A_{12}A_{22}^{-1}A_{12}^T & 0 \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} I & 0 \\ A_{22}^{-1}A_{12}^T & I \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12}A_{22}^{-1}A_{22} \\ A_{22}A_{22}^{-1}A_{12}^T & A_{22} \end{bmatrix}, \\ A_2 &= \begin{bmatrix} I & 0 \\ A_{12}^TA_{11}^{-1} & I \end{bmatrix} \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} - A_{12}^TA_{11}^{-1}A_{12} \end{bmatrix} \begin{bmatrix} I & A_{11}^{-1}A_{12} \\ 0 & I \end{bmatrix} = \begin{bmatrix} A_{11} & A_{11}A_{11}^{-1}A_{12} \\ A_{12}^TA_{11}^{-1}A_{11} & A_{22} \end{bmatrix}. \end{aligned} \tag{A.2}$$

The matrix A can be decomposed in A_1 if

$$A_{12} = A_{12}A_{22}^{-1}A_{22}, \tag{A.3}$$

or in A_2 if

$$A_{12} = A_{11}A_{11}^{-1}A_{12}. \tag{A.4}$$

The inverses of both the decompositions are given by

$$\begin{aligned} A_1^{-1} &= \begin{bmatrix} I & 0 \\ -A_{22}^{-1}A_{12}^T & I \end{bmatrix} \begin{bmatrix} (A_{11} - A_{12}A_{22}^{-1}A_{12}^T)^{-1} & 0 \\ 0 & A_{22}^{-1} \end{bmatrix} \begin{bmatrix} I & -A_{12}A_{22}^{-1} \\ 0 & I \end{bmatrix}, \\ A_2^{-1} &= \begin{bmatrix} I & -A_{11}^{-1}A_{12} \\ 0 & I \end{bmatrix} \begin{bmatrix} A_{11}^{-1} & 0 \\ 0 & (A_{22} - A_{12}^TA_{11}^{-1}A_{12})^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -A_{12}^TA_{11}^{-1} & I \end{bmatrix}. \end{aligned} \tag{A.5}$$

A.2 Matrix Inversion Lemma

If

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \tag{A.6}$$

the *Schur complement* of A_{11} is defined as

$$D_{11} = A_{11} - A_{12}A_{22}^{-1}A_{21}, \tag{A.7}$$

and the *Schur complement* of A_{22} as

$$D_{22} = A_{22} - A_{21}A_{11}^{-1}A_{12}. \tag{A.8}$$

The inverse of A can be written as

$$A^{-1} = \begin{bmatrix} A_{11}^{-1} + A_{11}^{-1}A_{12}D_{22}^{-1}A_{21}A_{11}^{-1} & -A_{11}^{-1}A_{12}D_{22}^{-1} \\ -D_{22}^{-1}A_{21}A_{11}^{-1} & D_{22}^{-1} \end{bmatrix}, \tag{A.9}$$

$$A^{-1} = \begin{bmatrix} D_{11}^{-1} & -D_{11}^{-1}A_{12}A_{22}^{-1} \\ -A_{22}^{-1}A_{21}D_{11}^{-1} & A_{22}^{-1} + A_{22}^{-1}A_{21}D_{11}^{-1}A_{12}A_{22}^{-1} \end{bmatrix}, \tag{A.10}$$

or as

$$A^{-1} = \begin{bmatrix} D_{11}^{-1} & -A_{11}^{-1}A_{12}D_{22}^{-1} \\ -A_{22}^{-1}A_{21}D_{11}^{-1} & D_{22}^{-1} \end{bmatrix}, \tag{A.11}$$

depending on whether $|A_{11}| \neq 0$, $|A_{22}| \neq 0$, or both (these can be verified by checking that $AA^{-1} = A^{-1}A = I$). By comparing the forms (A.9) - (A.11) the *matrix inversion lemma* is obtained

$$\left(A_{11}^{-1} + A_{12}A_{22}A_{21}\right)^{-1} = A_{11} - A_{11}A_{12}\left(A_{21}A_{11}A_{12} + A_{22}^{-1}\right)^{-1}A_{21}A_{11}. \tag{A.12}$$

A.3 Computation of the pseudo-inverse

The pseudo-inverse of a matrix is the matrix which gives the least squares solution to the matrix equation,

$$Ax = y, \tag{A.13}$$

when the matrix A is not square. It approximates the inverse of A and is therefore called the pseudo-inverse. The pseudo-inverse is computed as follows. Suppose the singular value decomposition (SVD) for a matrix $A : \mathbb{R}^m \to \mathbb{R}^n$ is

$$A = \begin{bmatrix} \vec{h_1}, & \vec{h_2}, & \ldots, & \vec{h_r} & \ldots & \vec{h_n} \end{bmatrix} \begin{bmatrix} s_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & 0 & s_r & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{a}_1 \\ \vec{a}_2 \\ \vdots \\ \vec{a}_r \\ \vdots \\ \vec{a}_m \end{bmatrix}, \tag{A.14}$$

were $s_1, \ldots, s_r$ are the non-zero singular values of the system. Ax is always in the column space of A, $Col(A)$, so if $y \notin Col(A)$, equation (A.13) will not have any solution. Therefore we need to solve $Ax = \bar{y}$ where $\bar{y}$ is the vector in $Col(A)$ closest to y, or the least squares solution.
Since $h_1, \ldots, h_r$ is an orthonormal base for $Col(A)$ we can compute this solution to be

$$\bar{y} = (h_1 \cdot y)h_1 + (h_2 \cdot y)h_2 + \ldots + (h_r \cdot y)h_r. \tag{A.15}$$

The following solution for x is obtained,

$$x = \frac{h_1 \cdot y}{s_1}a_1 + \frac{h_2 \cdot y}{s_2}a_2 + \ldots + \frac{h_r \cdot y}{s_r}a_r, \tag{A.16}$$

since

$$\begin{aligned} Ax &= A\left(\frac{h_1 \cdot y}{s_1}a_1 + \frac{h_2 \cdot y}{s_2}a_2 + \cdots + \frac{h_r \cdot y}{s_r}a_r\right) \\ &= \frac{h_1 \cdot y}{s_1}Aa_1 + \frac{h_2 \cdot y}{s_2}Aa_2 + \cdots + \frac{h_r \cdot y}{s_r}Aa_r \\ &= \frac{h_1 \cdot y}{s_1}s_1h_1 + \frac{h_2 \cdot y}{s_2}s_2h_2 + \cdots + \frac{h_r \cdot y}{s_r}s_rh_r \\ &= (h_1 \cdot y)h_1 + (h_2 \cdot y)h_2 + \cdots + (h_r \cdot y)h_r = \bar{y}. \end{aligned} \tag{A.17}$$

An easy way to compute (A.16) is using the matrix multiplication

$$x = \begin{bmatrix} a_1 & a_2 & \cdots & a_r \end{bmatrix} \begin{bmatrix} \frac{1}{s_1} & 0 & 0 & 0 \\ 0 & \frac{1}{s_2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \frac{1}{s_r} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_r \end{bmatrix} y. \tag{A.18}$$

The matrix $A^+ = \begin{bmatrix} a_1 & a_2 & \cdots & a_r \end{bmatrix} \begin{bmatrix} \frac{1}{s_1} & 0 & 0 & 0 \\ 0 & \frac{1}{s_2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \frac{1}{s_r} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_r \end{bmatrix}$ is called the pseudo-inverse of A and can be easily computed from the reduced SVD for a m x n matrix A.

Appendix B

Simulation model

In this thesis, simulations are performed on an example system. This system consists of two masses ($m_1 = 14.6496$ kg, $m_2 = 4.1856$ kg) coupled by a spring (with stiffness $k = 2 \cdot 10^3$ Nrad^{-1}) and a damper (with damping coefficient $d = 2$ Nrads^{-1}), the excitation force, u, is applied to the first mass, see figure B.1.

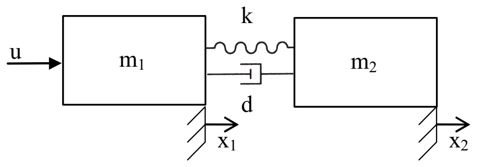


Figure B.1: Mass-spring-damper representation of the example system.

The equations which describe this system are

$$
\begin{aligned}
m_1\ddot{x}_1 &= -kx_1 - d\dot{x}_1 + kx_2 + d\dot{x}_2, \\
m_2\ddot{x}_2 &= kx_1 + d\dot{x}_1 - kx_2 - d\dot{x}_2.
\end{aligned}
\tag{B.1}
$$

When the state vector is chosen to be $x = \begin{bmatrix} x_1 & \dot{x}_1 & x_2 & \dot{x}_2 \end{bmatrix}^T$ and the output is $y = x_1$, the state-space representation of the system is

$$
\begin{aligned}
\dot{\mathbf{x}} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k}{m_1} & -\frac{d}{m_1} & \frac{k}{m_1} & \frac{d}{m_1} \\ 0 & 0 & 1 & 0 \\ \frac{k}{m_2} & \frac{d}{m_2} & \frac{-k}{m_2} & \frac{-d}{m_2} \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ \frac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} u, \\
y &= \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \mathbf{x},
\end{aligned}
\tag{B.2}
$$

The model is discretized at 1000 Hz and process and measurement noise are assumed to be present. The following linear time invariant discrete-time system which is used in the simulations is obtained

$$
\begin{aligned}
\mathbf{x}_{k+1} &= \begin{bmatrix} 3.9988 & -5.9969 & 3.9975 & -0.9994 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u_k + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} w_k, \\
y_k &= 10^{-6} \cdot \begin{bmatrix} 0.0682 & -0.1364 & 0.0682 & 0 \end{bmatrix} \mathbf{x}_k + v_k,
\end{aligned}
\tag{B.3}
$$

Bibliography

[1] B. D. O. Anderson and R. E. Skelton, "The Generation of all q-MArkov COVERs". *IEEE Trans. Autom. Control*, Vol. 35, 375–384.

[2] B. van Beek and B. de Jager, "An experimental facility for nonlineair robot control". *Proc. IEEE Internat. Conf. on Control Applications*, 1999, 668–673.

[3] B. van Beek and B. de Jager, "RRR-robot design: Basic outlines, servo sizing, and control". *Proc. IEEE Internat. Conf. on Control Applications*, 1997, 36–41.

[4] J.K. Bennighof, S.M. Chang, and M. Subramaniam, "inimum Time Pulse Response Based Control of Flexible Structures". *J. Guid., Control and Dyn.*, Vol. 16, No. 5, 1993, 874–881.

[5] B. Bukkems, "Advanced control of an RRR-robot tip". *Master's thesis*, Technische Universiteit Eindhoven, 2003, DCT 2003.04.

[6] M. C. Campi, A. Lecchini, and S. M. Savaresi, "Virtual Reference Feedback Tuning: A Direct Method for the Design of Feedback Controllers". *Automatica*, Vol. 38, 2002, 1337–1346.

[7] M. C. Campi, A. Lecchini, and S. M. Savaresi, "An application of the Virtual Reference Feedback Tuning Method to a Benchmark Problem". *Europ. J. Control*, Vol. 9, 2003, 66–76.

[8] K. Furuta, "Alternative solution of discrete-time Kalman filter". *Systems & Control Letters*, Vol. 22, 1994, 429–435.

[9] K. Furuta and M. Wongsaisuwan, "Closed-form solutions to discrete-time LQ optimal control and disturbance attenuation". *Systems & Control Letters*, Vol. 20, 1993, 427–437.

[10] A. Gelb, *Applied Optimal Estimation.* The M.I.T. Press, 1996.

[11] M. Gevers, "Identification for Control". *Proc. 5th IFAC Symposium on Adapt. Control Sig. Process*, 1995, 1–12.

[12] H. Hjalmarsson and M. Gevers, "Iterative Feedback Tuning: Theory and Applications". *IEEE Control Syst. Mag.*, Vol. 18, No. 4, 1998, 26–41.

[13] R. E. Kalman, "On the general theory of control systems". *Proc. 1st IFAC congr.*, Vol. 1, 1960, 481–492.

[14] D. Kostić, B. de Jager, M. Steinbuch and R. Hensen, "Modeling and Identification for Model-based Robot Control: An RRR-Robotic Arm Case Study". *Submitted to IEEE Trans. Control Syst. Tech.*, 2003.

[15] A. Lecchini, M. C. Campi, and S. M. Savaresi, "Virtual Reference Feedback Tuning for Two Degree of Freedom Controllers". *Internat. J. Adapt. Control Sig. Process*, Vol. 16, 2002, 355–371.

[16] F. L. Lewis and V. L. Syrmos, *Optimal Control.* John Wiley & Sons Inc., 1995.

[17] S. L. Marple, Jr., *Digital spectral analysis with applications.* Prentice Hall, 1987.

[18] The Mathworks Inc., *Using Matlab Revised for MATLAB 6 (Release 12)*, The Mathworks Inc., Natick, United States of America, November 2000.

[19] M. Q. Phan, R. K. Lim, and R.W. Longman, "Unifying Input-Output and State-Space perspectives of Predictive Control". *Technical report No. 3044*, Princeton University, Princeton, NJ, September 1998.

[20] R. E. Skelton and G. Shi, "Markov Data-Based LQG Control". *J. of Dyn. System, Meas. and Control*, Vol. 122, 2000, 551–559.

[21] M. Steinbuch and M. L. Norg, "Advanced Motion Control". *Europ. J. Control*, Vol. 4, No. 4, 1998, 278–293.

[22] R. L. Toussain, J. C. Boissy, M. L. Norg, M. Steinbuch, and O.H. Bosgra, "Suppressing Non-periodically Repeating Disturbances In Mechanical Servo Systems". *Proc. IEEE Conf. Dec. Control*,Tampa, Florida, 1998, 2541–2542.

[23] M. G. Safonov and T. C. Tsao, "The Unfalsified Control Concept and Learning". *IEEE Trans. Autom. Control*, Vol. 42, No. 6, 1997, 843–847.