Extended Kalman Filter based system identification tool

Extended Kalman Filter based System Identification Tool

R.H.A. Hensen

Report No. WFW 99.003

Eindhoven, January 1999

Eindhoven University of Technology
Department of Mechanical Engineering
Section Systems and Control

Extended Kalman Filter based System Identification Tool

Identification of nonlinear dynamic continuous-time models

For Use with Matlab[1] 5.2 or higher
Version 1.1

Ron Hensen, Email:ron@wfw.wtb.tue.nl

Technical Report No. WFW 99.003

Eindhoven, January 1999

Eindhoven University of Technology
Department of Mechanical Engineering
Section Systems and Control

[1]MATLAB is a trademark of The MathWorks, Inc.

Contents

Chapter 1

Introduction

In this report a combination of two worlds, i.e., (i) a theoretical and (ii) a tutorial part, will be presented. The theoretical part is a bases for the tool presented. Since optimal estimation theory, first presented by Kalman and others (circa 1960) [1], resulted in the extended kalman filter, a short introduction will be given on this theory. A larger contribution in this report will be the tutorials belonging to the toolbox.

1.1 Optimal Estimation

Before going into detail with respect to the Extended Kalman Filter (EKF) itself a general idea on optimal estimation is necessary. Gelb [1] gives a good definition for the optimal estimation

> *An optimal estimator is a computational algorithm that processes measurements to deduce a minimum error estimate of the state of a system by utilizing: knowledge of system and measurement dynamics, assumed statistics of system noises and measurement errors*

This idea can be depicted as in Figure 1.1 where $u(t)$ are the system inputs, $x(t)$ the system states, $z(t)$ the observations and $\hat{x}(t)$ the estimated system state. The optimal estimation enables

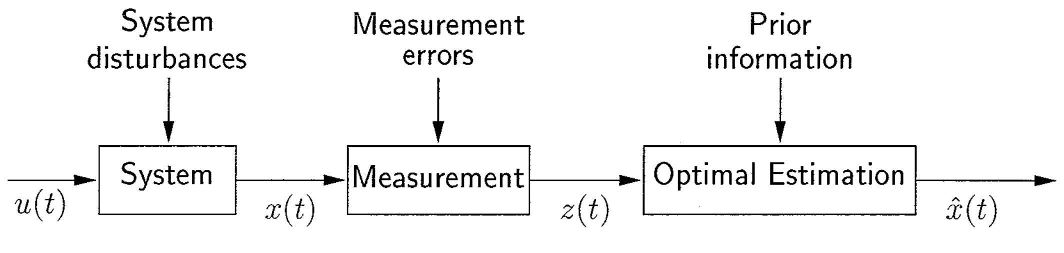


Figure 1.1: Optimal Estimation.

a sequential way of processing the measurement data, also called a recursive estimator. For both the system and measurements nothing is said about the class it belongs to. They might be classified with *discrete* or *continuous*, and *linear* or *nonlinear*.

Three different types of estimation tasks can be defined on the bases of optimal estimation theory, which are shown in Figure 1.2:

- Filtering, where an estimate is desired at the time instant of the last measurement point.
- Smoothing, where an estimate is desired at a time instant which falls within the span of available measurement data.
- Prediction, where an estimate is desired at a time instant after the last available measurement.

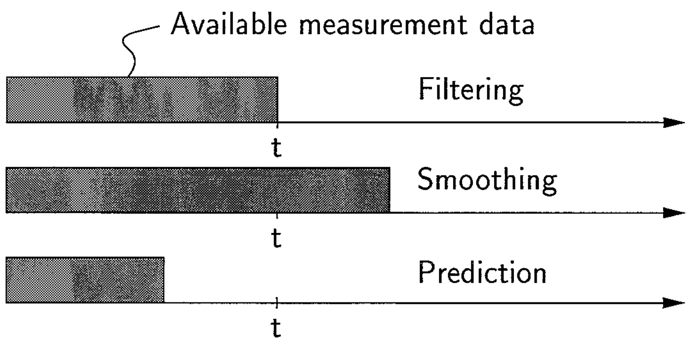


Figure 1.2: Three types of estimation tasks.

1.2 Tutorial

In the present toolbox, we focus on the filtering task for the identification of nonlinear dynamic continuous-time models. Here, the considered type of models can be written as

$$\dot{x} = f(x, u, t, \theta) \tag{1.1}$$
$$y = g(x, u, t, \theta) \tag{1.2}$$

where f and g are nonlinear vectorfields, x the model states, u the model inputs, y the model outputs, t the time and θ the model parameters. An additional objective in the context of prediction, i.e., simulation, will be presented to evaluate the accuracy of the identified models.

The objective of this toolbox is to present a methodology that is able to identify simulation models that yield accurate long-term prediction. The modelling procedure consists of two parts, i.e.,

- Identification of the model with limited available knowledge of the system to be modelled. This prior information must contain observations or measurements of the system. Additional *a priori* knowledge can be included in the choice of model structure.
- Validate to what extend the identified model is able to represent the system under consideration. This is done by simulating the system with the identified model and compare the real system response to the model response.

Both modelling parts can be performed by this toolbox which enables the engineer to quickly adjust model structures and to evaluate different identified models.

In Chapter 2 of this manual we present the Extended Kalman Filter (EKF) used in this toolbox. The adjustment of the filter to identify model parameters is given and additionally the filter parameters are explained. The model definitions and use of the toolbox in Matlab are discussed in Chapter 3. Two ways of using this toolbox are possible: (i) as a function which is used at the matlab prompt and (ii) via a Graphical User Interface which allows you to define and set the filter parameters. In Chapter 4, this manual will be closed by some examples which present the possible use of this toolbox.

Chapter 2

Extended Kalman Filter

In this chapter the Extended Kalman Filter (EKF) will be discussed and the procedure for the identification of model parameters is given. The EKF is based on optimal estimation for linear systems and extends the estimation for nonlinear dynamic stochastic systems as in (1.1) and (1.2) with

$$\begin{aligned} \dot{x} &= f(x,u,t) + w(t) && (2.1)\\ y &= g(x,u,t) + v(t) && (2.2) \end{aligned}$$

where f and g are nonlinear vectorfields, x the model states, u the model inputs, y the model outputs, t the time, $w(t)$ zero mean gaussian noise with a covariance matrix $Q(t)$ and $v(t)$ zero mean white noise with covariance matrix $R(t)$. Here, the class of estimation problems for nonlinear dynamic continuous-time systems with discrete-time measurements is considered. This reflects many practical situations where dynamic continuous-time processes are observed by measuring the system outputs at a certain sampling frequency. This type of estimation can be used in various engineering fields, e.g., mechanical, physical, electrical and chemical engineering.

The EKF is, like the Kalman Filter for linear systems, based on the minimization of the variance of the estimation error. The solution of this nonlinear minimum variance estimation problem as the minimum variance estimation problem (Kalman Filter) can be found in Gelb [1]. The resulting filter algorithm will be outlined below.

The filter algorithm is a recursive formulation that updates the estimates at discrete-time instants, i.e., when the outputs of the system are measured $m(t_i)$ $i = 1, 2, \ldots$. This update consists of the propagated state estimate $\hat{x}(t_i)$ corrected with a gain matrix $K(t_i)$ times the innovation signal $s(t_i)$

$$\begin{aligned} \hat{x}(t_i)^+ &= \hat{x}(t_i) + K(t_i)s(t_i)\\ &= \hat{x}(t_i) + K(t_i)\left[m(t_i) - g(\hat{x},u,t_i)\right] && (2.3) \end{aligned}$$

where $+$ denotes the new update and $\hat{}$ is the state estimate. The gain matrix K is chosen to minimize the variance of the reconstruction error ε, i.e., $\varepsilon = x - \hat{x}$. At each discrete-time instant the gain matrix is computed by

$$K(t_i) = P(t_i)G^T(\hat{x},u,t_i)\left[G(\hat{x},u,t_i)P(t_i)G^T(\hat{x},u,t_i) + R(t_i)\right]^{-1} \qquad (2.4)$$

and the error covariance matrix $P(t_i)$ is updated with

$$P(t_i)^+ = [I - K(t_i)G(\hat{x}, u, t_i)]\, P(t_i) \tag{2.5}$$

Between two discrete-time instants t_i and t_{i+1} the state estimate and error covariance are propagated by integration of

$$\begin{aligned} \dot{\hat{x}} &= f(\hat{x}, u, t) && (2.6)\\ \dot{P}(t) &= F(\hat{x}, u, t)P(t) + P(t)F^T(\hat{x}, u, t) + Q(t) && (2.7) \end{aligned}$$

Here, as stated before, are the model errors $v(t)$ and $w(t)$ considered to be zero mean gaussian noise having a spectral density matrices $Q(t)$ for the state errors and $R(t)$ for the measurement errors. Furthermore, are the state errors and measurement errors assumed to be uncorrelated. The matrices $F(\hat{x}, u, t)$ and $G(\hat{x}, u, t)$ are Jacobian matrices of the nonlinear model functions $f(x, u, t)$ and $g(x, u, t)$ with respect to x which are evaluated at $x = \hat{x}$. For the initial state estimates $\hat{x}(t_0)$ our confidence or error variance can be expressed by the initial diagonal covariance matrix $P(t_0)$. Diagonal elements unequal to zero express uncertain initial state estimates where zero elements express infinite confidence.

The nonlinear parameter estimation for general nonlinear dynamic continuous-time models as described in (1.1) and (1.2) can be transformed in a state reconstruction problem by defining the *augmented state*

$$x_{aug} = \begin{bmatrix} x \\ \theta \end{bmatrix}$$

Consequently, Eq. (1.1) has to be augmented with an additional k differential equations $\dot{\theta} = 0$, where k is the number of model parameters. Hence, the model parameters are considered as constants. This state reconstruction problem can be solved by the EKF as discussed above.

Due to computational and practical convenience the EKF is used off-line in this toolbox. This results in the situation where only a finite number of measurements of the system inputs and ouputs can be used for the identification of the models. Since this finite number of measurements is also the maximal number of filter estimates, the parameter estimates of the augmented state might not be converged to constants yet. In the situation where the model parameters are converged to constants the filter parameter estimates become smoothed parameter estimates which implies that effectively an output error criterion is minimized. One way to deal with this limited system information is to design an iterative EKF where the data is passed through the filter several times untill the parameter estimates are converged. After each filter pass, the initial state estimates $\hat{x}(t_0)$ and the corresponding confidence $P(\hat{x}(t))$ are re-initialized with the initial estimates of the first pass. The parameters θ and the corresponding covariance matrix $P(\hat{\theta}(t))$ are reset to the final estimates of the previous filter pass.

Chapter 3

Model definitions and (GUI) System Identification Toolbox

In this chapter the installation and usage of the toolbox will be discussed. This version 1.1 is written for MATLAB 5.2 or higher. The version has been tested under MATLAB 5.2 for WINDOWS[1] 95 on an IBM compatible PENTIUM.

3.1 Installing the toolbox

The toolbox consists of two zip-files, i.e.,

- ekf.zip which is actually the toolbox itself. This file should be copied and unzipped to one directory. Additionally it is necessary that this directory is added to the matlabpath.
- examples.zip consists of two examples which will be explained in Chapter 4. For each example the model definition files are given together with the corresponding system input/output measurements. This file can be unzipped either in the toolbox-directory or in another matlabpath-directory.

All files, compressed in the zip-files, are compatible for use with UNIX or WINDOWS versions of MATLAB 5.2 or higher.

3.2 Using the toolbox

The toolbox is built around the matlab function ekf which is actually the Extended Kalman Filter algorithm. A Graphical User Interface started with ekf_gui enables the user to specify all filter settings graphically. The identification or validation can be started from this interface. The model functions f and g can be defined in **one** M-file. A template M-file is available, i.e., template.m. On the following pages these different matlab functions and M-files will be described as in the Matlab manuals.

[1]MS-WINDOWS is a trademark of MICROSOFT CORPORATION

ekf

Purpose

Solve Extended Kalman Filter Algorithm

Syntax

```
[Xaug,Ymod,Trace] = ekf(x0,Theta0,U,Y,T,model,R,Q,P,...
                        passes,order,s,inverse)
```

Arguments

x0	Initial state estimates $\hat{x}(t_0)$.
Theta0	Initial model parameter estimates $\hat{\theta}(t_0)$.
U	System input measurements. Size: [# of inputs, # of measurements]
Y	System output measurements. Size: [# of outputs, # of measurements]
T	Discrete time instants of measurements. Size: [1, # of measurements]
model	String corresponding to M-file defining the model functions f and g.
R	Variance matrix of measurements. The elements specify the diagonal elements of the matrix $R(t)$. Here, these variances are considered to be constants. Hence, every output measurement over time of each output is equally weighed. Size: [# of outputs, 1]
Q	Variance matrix of augmented state errors. The elements specify the diagonal elements of the matrix $Q(t)$. Here, these variances are considered to be constants. Size: [# of augmented states, 1]
P	Variance matrix of initial augmented state estimates. The elements specify the diagonal elements of the matrix $P(t_0)$. Size: [# of augmented states, 1]
passes	Number of filter passes.
order	The input order hold during propagation of $\hat{x}(t)$. 0 specifies zero order hold and 1 first order hold.
s	String correpsonding to the integration scheme used. Example: s = 'ode23tb'. See help ode23 and others.
inverse	Inversion of gain matrix. 0 specifies normal inversion and 1 the pseudo inversion.

template

Purpose

Define $f(x_{aug}, u, t)$ and $g(x_{aug}, u, t)$ model functions.

Description

template is not a command or function. It is a help entry how to create an M-file defining the model functions f and g to be used in the EKF algorithm.

You can use the template M-file to define model equations of the following form:

$$\dot{x} = f(x_{aug}, u, t)$$

where:

t	is a scalar independent variable, it represents time.
x	is a vector of dependent variables, it represents the model state.
x_{aug}	is a vector of dependent variables, containing model state x and model parameters θ.
u	is a vector of dependent variables, it represents the model inputs.
f	is a function of t, u and x_{aug} returning a column vector the same length as x.

Additionally, it is necessary to give the Jacobian of f with respect to x_{aug}. This is defined in the matrix F

$$F = \frac{\partial f(x_{aug}, u, t)}{\partial x_{aug}}$$

which should be a matrix of size:

[# of model states, # of augmented states].

You can use the template M-file to define model equations of the following form:

$$y = g(x_{aug}, u, t)$$

where:

t	is a scalar independent variable, it represents time.
x_{aug}	is a vector of dependent variables, containing model state x and model parameters θ.

u	is a vector of dependent variables, it represents the model inputs.
y	is a function of t, u and x_{aug} returning a column vector the same length as the model output.

Additionally, it is necessary to give the Jacobian of g with respect to x_{aug}. This is defined in the matrix G:

$$G = \frac{\partial g(x_{aug}, u, t)}{\partial x_{aug}}$$

which should be a matrix of size:

[# of model outputs, # of augmented states].

In this file are some restricted areas for the user. These areas started with a !!!-line should not be edit by the user. Only areas started with a —-line should be edit by the user.

To use the template file:

- Enter the command `help template` to display the help entry.
- Cut and paste the template file text into a separate file.
- Edit the file to your problem.

ekf_gui

Purpose Graphical User Interface of EKF algorithm.

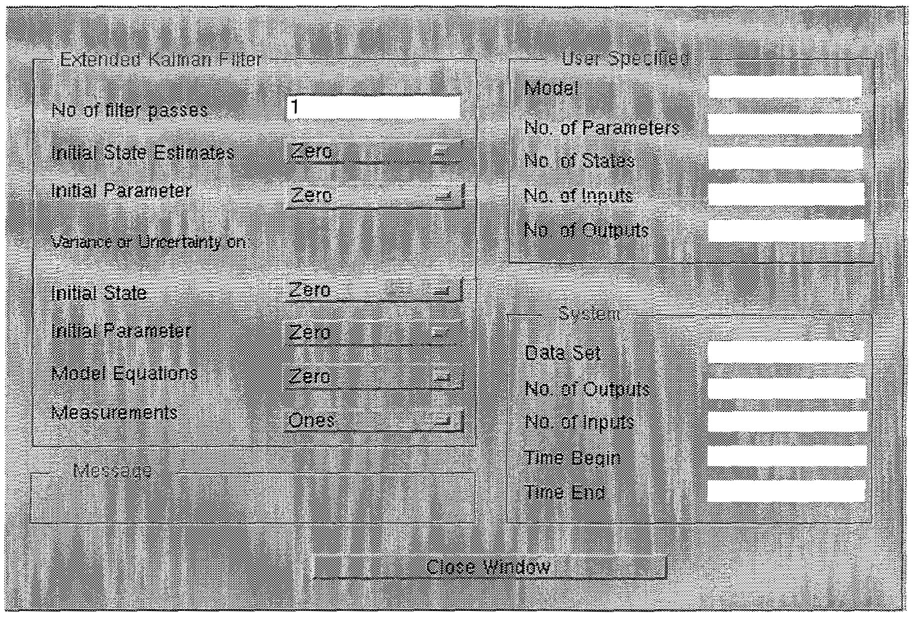
Syntax ekf_gui

Description This Graphical User Interface enables the user to specify all the function inputs of ekf. The GUI has five menu items with each different subfunctions. The different menu items will be discussed below:

- Load
  - Model
  - Data set T, U and Y
  - Identification results
  - Validation results
- Save
  - Identification results
  - Validation results
- Start
  - Identification
  - Validation
  - Trial identification
- Show
  - Data set
  - Identification
  - Validation
- Advanced
  - Integration Scheme
  - Input order hold
  - Inverse

Most of these subfunstions are trivial and are discussed at the definition of the function ekf. The models that are loaded are represented in the GUI on the right hand side. Also is the name of the loaded data set given with additional information. On the left hand side of the GUI the filter parameters can be entered graphically. On the next page a screenshot of this GUI is given with the distinction between the right and left side. The message box on the left side gives information when incorrect actions are performed or other improper definitions are detected.

ekf_gui



Extended Kalman Filter

No of filter passes: 1

Initial State Estimates: Zero

Initial Parameter: Zero

Variance or Uncertainty on

Initial State: Zero

Initial Parameter: Zero

Model Equations: Zero

Measurements: Ones

Message

User Specified

Model

No. of Parameters

No. of States

No. of Inputs

No. of Outputs

System

Data Set

No. of Outputs

No. of Inputs

Time Begin

Time End

Chapter 4

Examples

4.1 Two-tank system

In order to illustrate the use of the EKF technique, a simulation study is performed on a two-tank system. The two-tank system is given in Fig. 4.1 and Table 4.1. For the two-tank system, the differential equations are

$$A_1\dot{h}_1 = -k_1\sqrt{h_1} + F_1 \tag{4.1}$$

$$A_2\dot{h}_2 = -k_2\sqrt{h_2} + k_1\sqrt{h_1} + F_2 \tag{4.2}$$

Due to the nonlinear right hand side terms of the Eqs. 4.1, 4.2, e.g., $\sqrt{h_1}$, $\sqrt{h_2}$, the two-tank system is nonlinear.

Symbol	Value	Unit	Description
F_1	0-0.2	l/s	Flow, control input
F_2	0-0.1	l/s	Flow, control input
F		l/s	Flow out
h_1	0-1	dm	Height in tank 1
h_2	0-0.5	dm	Height in tank 2
A_1	10	dm^2	Area in tank 1
A_2	5	dm^2	Area in tank 2
k_1	0.0791	$(\mathrm{l/s})\mathrm{dm}^{-1/2}$	Flow parameter tank 1
k_2	0.1342	$(\mathrm{l/s})\mathrm{dm}^{-1/2}$	Flow parameter tank 2
LS			Level sensor

Table 4.1: Definition of symbols for the two-tank system.

This two-tank system is used to obtain datasets for the identification of the model proposed in the remainder of this section. This is an ideal case where the model structure is exactly known, since the structure of the system, i.e., Eq. 4.1 and 4.2, is exactly known and the measured outputs are free of noise.

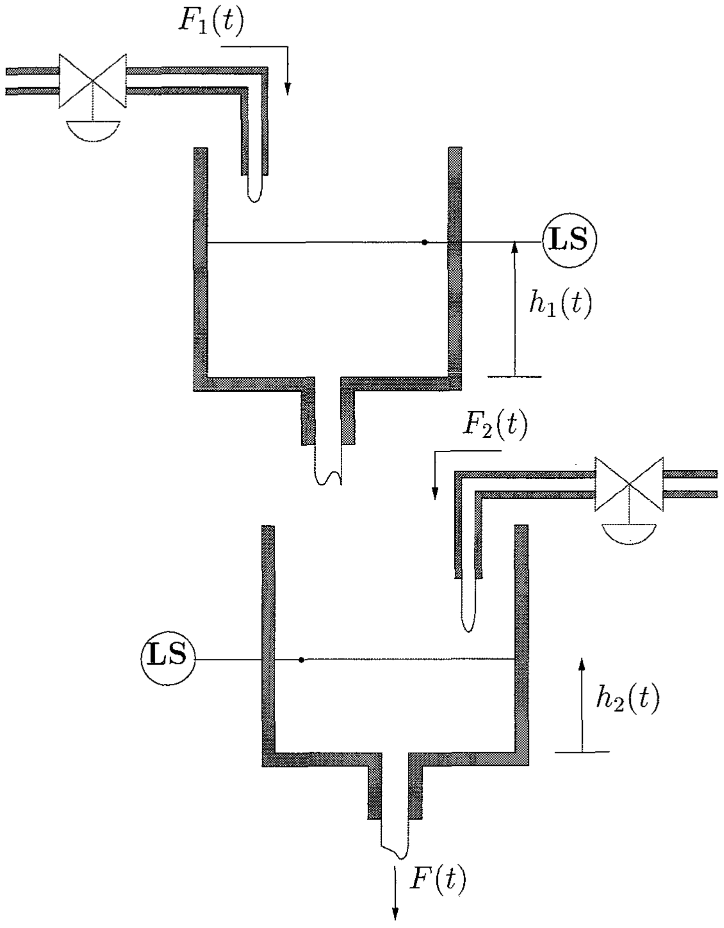


Figure 4.1: Two-tank system.

The model structure is chosen equal to the Eqs. 4.1 and 4.2 which results for the model differential equations in

$$\begin{aligned} \theta_1 \dot{x}_1 &= -\theta_3 \sqrt{x_1} + F_1 && (4.3) \\ \theta_2 \dot{x}_2 &= -\theta_4 \sqrt{x_2} + \theta_3 \sqrt{x_1} + F_2 && (4.4) \end{aligned}$$

where the parameters to be identified are

$$\theta = [A_1 \ \ A_2 \ \ k_1 \ \ k_2]^T$$

The tank heights are measured, so the model outputs become:

$$\begin{aligned} y_1 &= x_1 && (4.5) \\ y_2 &= x_2 && (4.6) \end{aligned}$$

The model is given in the M-file Two_tank_model.m and the measured input-ouput dataset in the MAT-file Two_tank.mat which is shown in Figure 4.2.

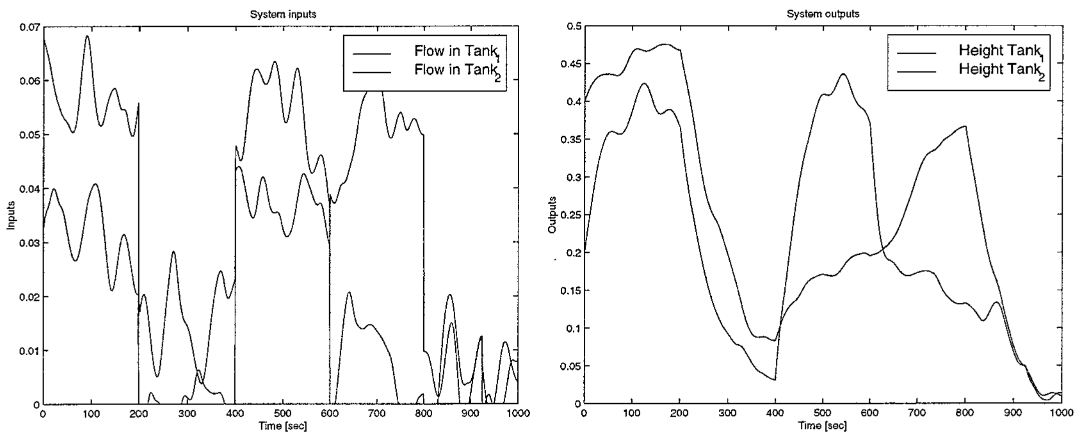


Figure 4.2: Dataset corresponding to two-tank system.

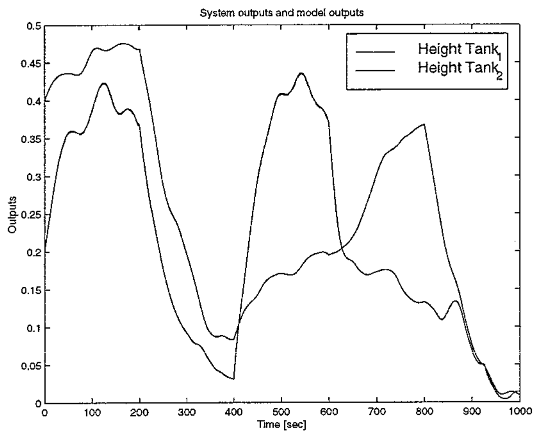
The system is identified with the following settings:

- Filter passes: 10
- Initial state estimates: $\hat{x}_0 = Y(t_0)$
- Initial parameter estimates: $\hat{\theta}_0 = [20\ 20\ 1\ 1]^T$
- Covariance on initial state: $P(\hat{x}_0) = [0\ 0]^T$
- Covariance on initial parameters: $P(\hat{\theta}_0) = [10\ 10\ 10\ 10]^T$
- Covariance on model equations: $Q = [0\ 0]^T$, because of exact modeling
- Covariance on measurements: $R = [0.001\ 0.001]^T$

The tuning of the EKF parameters is nontrivial and often based on trial and error. After 10 filter passes the parameter estimates have indeed become constant and the average parameter values over the last filter pass are used to validate the identified model. The average parameter estimates of the last filter pass are

$$\hat{\theta}_{10} = [9.9579\ 5.0238\ 0.07904\ 0.13439]^T$$

and are close to the true values as given in Table 4.1. The identification results are given in the MAT-file Id_two_tank.mat and the validation results in the MAT-file Val_two_tank.mat. The validation results are shown Figure 4.3, where the difference between the system outputs and model outputs are not observable in the left figure. The results are very good since the estimated parameters are very close to the true values.



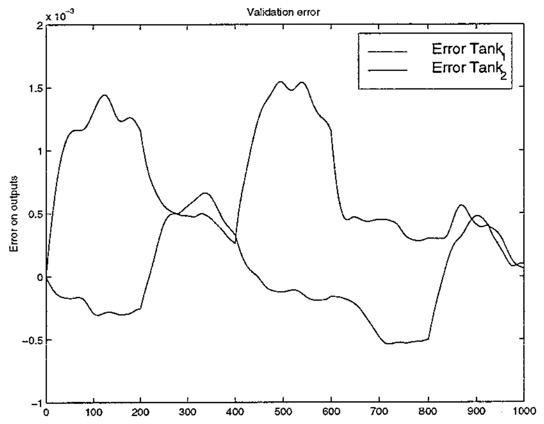


Figure 4.3: Validation results.

4.2 Mechanical System with Friction

Another example for the use of the EKF algorithm is a study where in contrary to the previous example a practical lab-scale system is used. Hence, this example is less ideal and involves nonexact modeling and measurement noise.

The rotating arm in Fig. 4.4 is a nonlinear mechanical system [2] with one degree of freedom q, where q represents the angular displacement of the arm. The state-space equations are

$$\frac{d}{dt}\begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \dot{q} \\ -M^{-1}(\theta)C(\dot{q},\theta) \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}(\theta)c_m \end{bmatrix} u \tag{4.7}$$

where $C(\dot{q},\theta)$ represents friction, $M(\theta)$ is the effective inertia of the motor-transmission-rotating arm combination, c_m is the motor gain and u is the motor input current. We assume that

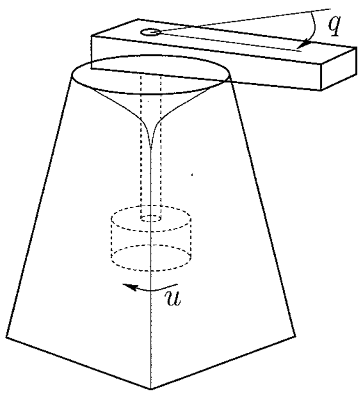


Figure 4.4: Rotating arm.

the friction torque $C(\dot{q},\theta)$ is a nonlinear function of the angular velocity $\dot{q}$ and of the model parameters θ. Here, it is assumed that the friction can be modeled by an odd continuous function

$$\hat{C}(\dot{q},\theta) = -\hat{C}(-\dot{q},\theta) \qquad \dot{q} \in \mathbb{R} \tag{4.8}$$

For angular velocity equal to zero the friction model torque is zero, which results for the model in the same set of equilibrium points as for the system.

Here, the friction will be modeled with a neural network. The neural network consists of two layers, i.e., one hidden layer and one output layer. The neural network represents a nonlinear mapping from the network input $\mathbb{R}^r$ into the network output $\mathbb{R}^s$. Here, this mapping is from angular velocity $\dot{q}$ ($\mathbb{R}$) to friction model torque $\hat{C}(\dot{q}, \theta)$ ($\mathbb{R}$). Defining the weight matrices for the first and second layers as W_1 and W_2, one can write the neural network output as

$$\hat{C}(\dot{q}, \theta) = W_2^T \Sigma(W_1 \dot{q} + b_1) + b_2$$

where b_i represents the bias value for the neurons in the i-th layer and $\Sigma(.)$ is a nonlinear operator with $\Sigma(\underline{z}) = [\sigma(z_1), \ldots, \sigma(z_v)]^T$, $\sigma(.)$ a differentiable, nonlinear, monotonic increasing function and v is the number of hidden neurons.

To assure the system properties described above to hold the following restrictions are posed on the neural network topology

- Choose an odd function for $\sigma(.)$ which is equal to zero if its argument is zero.

$$\sigma(z_i) = 1 - \frac{2}{e^{2z_i} + 1}$$

- The first choice together with the set of equilibrium points for the system implies that the bias terms should be zero.

These two restrictions result for the neural network friction approximator in

$$\hat{C}(\dot{q}, \theta) = W_2^T \Sigma(W_1 \dot{q})$$

The linear part of the system dynamics, i.e., the viscous damper characteristic and the input term are modeled as in Eq. 4.4. In state space description the model becomes

$$\begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{\hat{b}}{\hat{M}} \end{bmatrix} \begin{bmatrix} q \\ \dot{q} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{W_2^T \Sigma(W_1 \dot{q})}{\hat{M}} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{c_m}{\hat{M}} \end{bmatrix} u \tag{4.9}$$

where $\hat{b}$ is the viscous damper constant of the system. For the model parameters this results in

$$\theta = [W_1^T \; W_2 \; \hat{b} \; \hat{M}]^T$$

Since the angular position q and angular velocity $\dot{q}$ are measured, the model outputs become

$$y_1 = q \tag{4.10}$$

$$y_2 = \dot{q} \tag{4.11}$$

The model is given in the M-file Robot.m, where the Neural Network consists of 3 neurons. Hence, W_1 and W_2 contain each three parameters setting the total number of parameters to be identified to 8. The measured input-ouput dataset in the MAT-file Robot.mat which is shown in Figure 4.5.

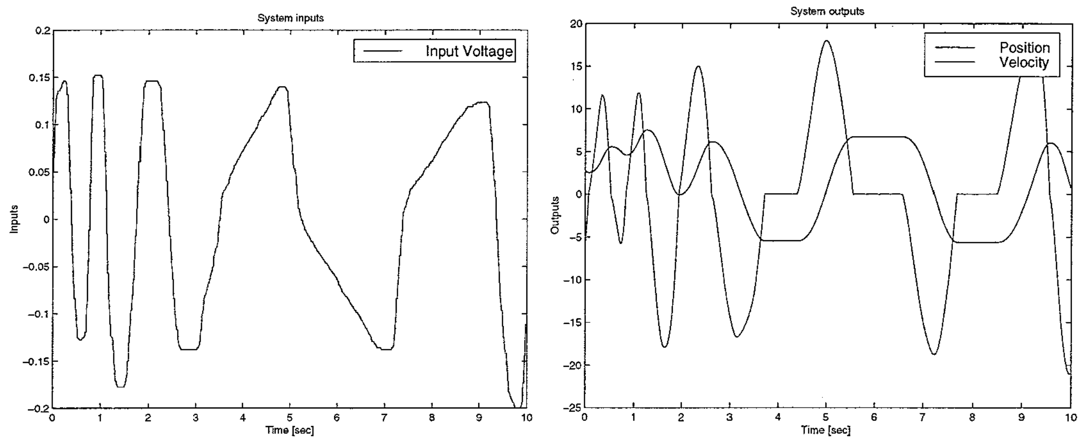


Figure 4.5: Dataset corresponding to Mechanical system.

The system is identified with the following settings:

- Filter passes: 15
- Initial state estimates: $\hat{x}_0 = Y(t_0)$
- Initial parameter estimates: $\hat{\theta}_0 = [1\ 1\ 1\ 1\ 1\ 1\ 0\ 10]^T$
- Covariance on initial state: $P(\hat{x}_0) = [0\ 0]^T$
- Covariance on initial parameters: $P(\hat{\theta}_0) = [1\ 1\ 1\ 1\ 1\ 1\ 10\ 10]^T$
- Covariance on model equations: $Q = [0\ 0.001]^T$, because of nonexact modeling
- Covariance on measurements: $R = [0.0001\ 0.001]^T$

After 15 filter passes the parameter estimates have indeed become constant and the average parameter values over the last filter pass are used to validate the identified model. The average parameter estimates of the last filter pass are

$$\hat{\theta}_{15} = \begin{bmatrix} 0.2069 \\ 12.8021 \\ 0.0428 \\ 1.3460 \\ -1.5092 \\ -0.0355 \\ -0.0778 \\ 34.4013 \end{bmatrix}$$

The identification results are given in the MAT-file Id_robot.mat and the validation results in the MAT-file Val_robot.mat. The validation results are shown Figure 4.6, where in the left figure the dashed lines are the model outputs and the solid lines are the system outputs.

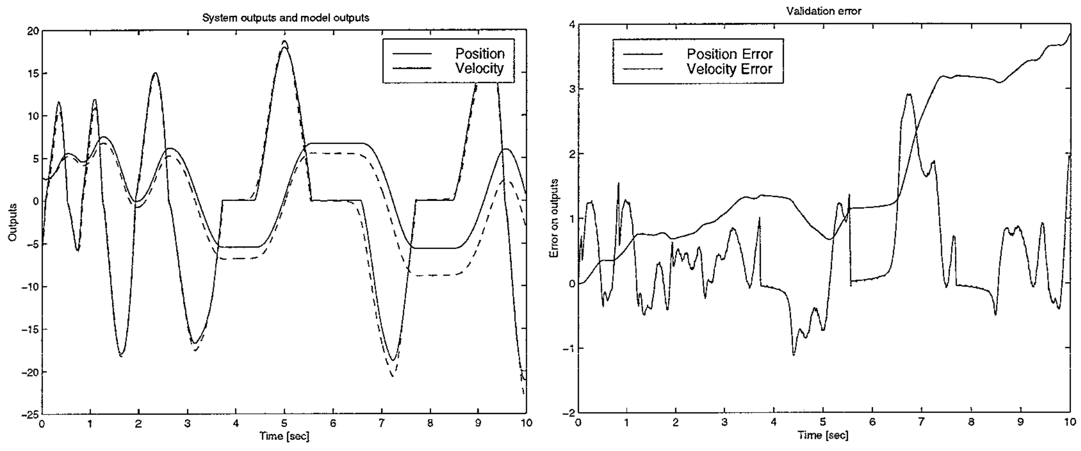


Figure 4.6: Validation results.

These results can be interpreted as follows: (i) the inertia $\hat{M} = 0.0291[kg.m]$ and (ii) the friction curve $\hat{C}(\dot{q}, \theta)$ together with the viscous damping $\hat{b}\dot{q}$ gives a curve as depicted in Figure 4.7.

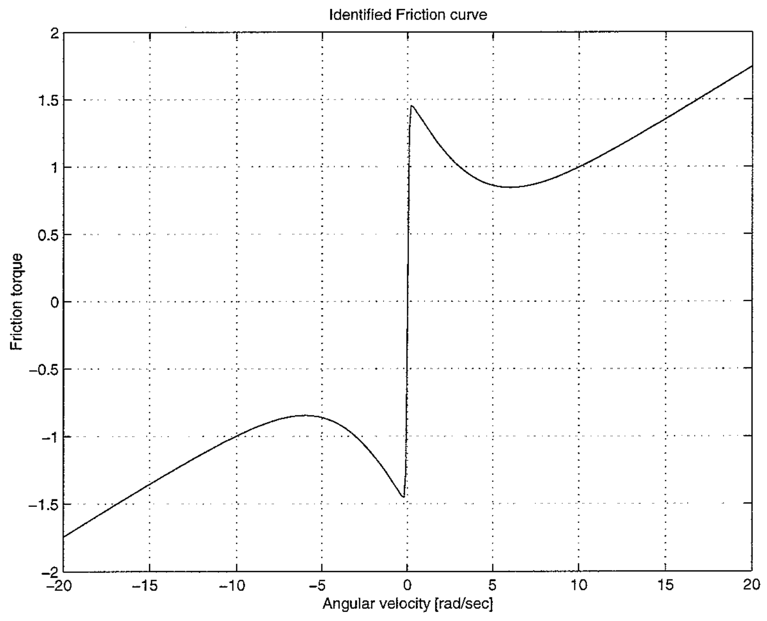


Figure 4.7: Identified Friction curve.

Bibliography

[1] A. Gelb. *Applied optimal estimation.* M.I.T. Press, 1978.

[2] H. Nijmeijer and A.J. van der Schaft. *Nonlinear Dynamical Control Systems.* Springer, 1990.