HELSINKI UNIVERSITY OF TECHNOLOGY
Control Engineering Laboratory

KALMAN FILTERING IN MULTI-SENSOR FUSION

Tibor Digaňa

Master's Thesis for the degree of Master of Science in Technology
September 2004

Supervisor	Professor Heikki Koivo	
Instructor	Vesa Hasu, Lic. Tech	VESA HASU

Helsinki University of Technology
Department of Automation and Systems Technology
The Control Engineering Laboratory

Create by TIBOR DIGAŇA http://www.hut.fi/u/digana/
http://users.tkk.fi/~digana/

Data CD is enclosed to this master's thesis on web-page http://www.hut.fi/u/digana/MT.html

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY

HELSINKI UNIVERSITY OF TECHNOLOGY
CONTROL ENGINEERING LABORATORY
P.O. Box 5500
FIN-02015 HUT
http://www.hut.fi http://www.tkkt.fi

ABSTRACT OF MASTER'S THESIS

Author	Tibor Digaňa tibor.digana@hut.fi
Topic of the thesis	Kalman filtering in multi-sensor fusion
Date of manuscript	September 17, 2004
Date of given presentation	12 – 14pm, August 16, 2004
Department	Automation and Systems Technology
Laboratory	The Control Engineering Laboratory
Field of study	Control Engineering
Opponent(s)	
Supervisor	Professor Heikki Koivo
Instructor	Vesa Hasu, Lic. Tech

Abstract

This thesis answers several questions of decentralized Kalman-Filters in multi-sensor fusion, fault detection and isolation in sensors, optimal control in linear-quadratic Gaussian problem, an algorithm in fuzzy based approach to adaptive Kalman-Filtering additionally in multi-state multi-sensor fusion. Generally, Kalman-Filters comprise a number of types and topologies depending on use and computing complexity of applied processors. State estimation provided by a Kalman-Filter is crucial in this thesis.

In 1960, R.E. Kalman published his famous paper describing a recursive solution to the discrete-data linear filtering. His developed Kalman-Filter performs optimal estimation of an unknown system state through filters behaviour. This thesis supposes some models of promising linear Kalman-Filter simulated beyond MATLAB and Simulink program especially utilised in the fields of steering-controls or navigations, etc.

The script of thesis consists of two parts, namely:

Techniques. The first part presents some techniques in Kalman-Filtering with a variety of Kalman-Filters such as a conventional estimator used for simple single node topology (such as CoKF), a fully decentralized, DKF, and centralized, CKF, Kalman-Filters and their appropriate topologies.

Simulatons. The second part deals with some tests of affected state estimators such as decentralized Kalman-Filter, and test of state estimation identity of centralized Kalman-Filter with decentralized Kalman-Filter. In addition to Kalman-Filtering, some implementations of these estimator kinds are used in some simulations of regulator in linear-quadratic Gaussian problem; fuzzy adaptive Kalman-Filtering; and some tests of traditional fault detection and isolation algorithm in decentralized topology of sensors.

Keywords	Kalman-Filtering, multi-sensor fusion, optimal control, fuzzy computing
Number of pages	128
Print distribution	Helsinki University of Technology, The Control Engineering Laboratory

The thesis can be read at http://www.hut.fi/u/digana/Doc/MT.pdf more on www.hut.fi/u/digana/MT.html

Preface

Firstly, I am truly indebted to my supervisory, Professor Heikki Koivo, and instructor, Vesa Hasu, who have guided me through the whole of my work. I have learned quite a lot from their comments and humour being always fruitful.

Heikki's humour, like "Aaaa, we are shouting! – on May of teacher's celebration day"

"Especially, when, … but …" At the leaving party, " I'm not the 5 years ahead predictor!"

First time I joined Helsinki University of Technology as an exchange student on the January, 2002. I started undergraduate courses and I got a wide knowledge in fuzzy logic controllers, neural networks, genetic algorithms, radio network planning methods, audio and video signal processing. It was very good experience to me during the spring term. That time, Professor Heikki Koivo as my teacher inspired me with a possibility to make a master's thesis.

Then, I definitely decided for HUT and did not want settle for less – another university.

After the rest in Slovakia, I started my Master's Thesis at the control laboratory of HUT in Finland the 15th of December 2004. The field of this thesis, the author was working on, was decided due to the most exciting area of research seemed to me in digital signal processing. I acquired the student's status with the help of Heikki Koivo after over one year of effort. So, Professor Heikki Koivo provided me with the invaluable opportunity enabling me to come to Finland.

I express my warm Thanks to the control laboratory staff and Tarja Timonen for their kind assistance in solving problems, that a foreign student was dealing with.

My accommodation problems were immediately done away with the thanks to the kind offer of Center for International Mobility (CIMO) organization.

I address my appreciation of encouragement to my dear parents, mother Oľga and my father Vladimír.

My unforgotten Thanks go to all friends of mine in Finland and Slovakia.

I would like to say great Thanks to God who guides me, I believe, through my life forever.

I received a scholarship from The Control Engineering Laboratory, Helsinki University of Technology in Finland. On September, 23, 2004, I graduated at University of Žilina in Slovaka and I succeed with this great master's thesis that was undertaken abroad.

Contents

Kalman-Filter Techniques

MATLAB Simulations and Tests

List of abbreviations

ADKF	Adaptive Decentralized Kalman Filter
AKF	Adaptive Kalman Filter
AKf	Adaptive Kalman filtering
CKF	Centralized Kalman Filter
CoKF	Conventional Kalman Filter
DKF	Decentralized Kalman Filter
EKF	Extended Kalman Filter
FD	Fault Detection method
FDI	Fault Detection and Isolation
FI	Fault Isolation method
FIS	Fuzzy Interface Syetem
GAs	Genetic Algorithms
IAE	Innovation Adaptive Estimation
KF	Kalman Filter
LQR	Linear Quadratic Regulator
LQG	Linear Quadratic Gaussian problem
MSE	Mean Square Error
NNRx	Noise to Noise Ratio measured in states of plant
NNRy	Noise to Noise Ratio measured in observations of plant
PCA	Principle Component Analysis
PSD	Power Spectral Density

List of symbols

$\mathbf{A}$	State transition matrix; squared matrix, in discrete time domain
$\mathbf{A_c}$	State transition matrix of system in closed-loop linear-quadratic regulator; squared matrix, in discrete time domain
$\mathbf{a}(n)$	Auxiliary variable used in some substitutions, defined locally
$\mathbf{As}(n)$	Steady state transition matrix; squared matrix, in discrete time domain
$\mathbf{B}$	Input coupling vector of input model; column vector, in discrete time domain
$\mathrm{BinaryCorr}_{ij}(n)$	Coefficient of hard limited cross-correlation function applied to FDI system, the bitwise operand and single value, in discrete time domain
$\mathbf{C}$	Observation or sensitivity vector (observation function in EKF); (row vector in CoKF); (matrix in CKF, DKF, AKF, ACKF, ADKF), in discrete and continuous time domain
$\mathbf{Cs}(n)$	Steady observation vector; (only in Model 3 of KF), in discrete time domain
$\mathrm{Corr}_{ij}(n)$	Cross-correlation coefficients applied to FDI system, in discrete time domain
$d(n)$	Drift function of second observation in a system, in discrete time domain
$d_1(n)$	Drift function of first observation in a system, in discrete time domain
$\mathrm{Enable}_{ij}(n)$	Enable signal applied to FDI system, in discrete time domain
$\mathbf{F}$	Dynamic coefficient matrix or simply the dynamic matrix and its elements are called the dynamic coefficients of a system; squared matrix, in continuous time domain
$\mathbf{G}$	Process noise coupling vector of a system; column vector, in continuous time domain
$\mathbf{I}$	Identity matrix
i, j, k	Subscripts of a sum, or subscripts of a matrix elements, or index denoting an iteration
J	Performance index, or quadratic performance criterion (index), or cost function
$\mathbf{K}$	Feedback gain in discrete-time linear quadratic regulator; row vector
K_f	Feed-forward gain in discrete-time linear quadratic regulator
L	Output fuzzy set
l_o	Order (length) of state vector related to identity matrix of a noise
$\mathbf{M}(n)$	Innovation Kalman filter gain; column vector, in discrete time domain
N	Normal distribution function
N_o	Number of sensors in Kalman filters

$\mathrm{NCorr}_{ij}(n)$	Related (normalized) cross-correlation coefficients applied to FDI system, in discrete time domain
n	Discrete time
$\mathbf{O}_{ii}$	Elements of a noise identity matrix; squared matrix
$\mathbf{O}$	Identity matrix of a noise; squared matrix
$\mathbf{o}_{ii}$	The *ii*-th element of a matrix of normal probability distribution function
$\mathbf{P}(n+1\|n)$	Time update of state error covariance matrix; squared matrix, in discrete time domain
$\mathbf{P}(n\|n)$	Estimate update of state error covariance matrix; squared, in discrete time domain
$\mathbf{Ps}(n)$	Steady state error covariance matrix; squared matrix, in discrete time domain
$\mathbf{Q}(n)$	Covariance squared matrix of process noise, in discrete time domain
$\mathbf{Q}_{\mathbf{a}}(n)$	Adapted covariance squared matrix of process noise, in discrete time domain
$\mathbf{Q}_{\mathbf{d}}(n)$	Defaulted covariance diagonal matrix of process noise, in discrete time domain
$\mathbf{Q}_{\mathbf{global}}(n)$	Process noise covariance matrix statistically combined from all nodes in adaptive Kalman filtering of multi-sensor fusion (ACKF, ADKF), in discrete time domain
$\mathbf{R}(n)$	Observation (sensor) noise covariance, in discrete time domain
$R_a\ (n)$	Adapted observation (sensor) noise variance, in discrete time domain
$\mathrm{RS}_{ij}(n)$	Reference signal applied to FDI system, in discrete time domain
$r(n)$	Innovations sequence (residuals), in discrete time domain
$\mathbf{S}(n)$	Associated noise, in discrete time domain
$\mathbf{SEI}(n)$	State error information in decentralized Kalman filter; column vector, in discrete time domain
$\mathbf{U}(n)$	Cost-weighting (state penalty) matrix in discrete-time linear-quadratic regulator
$u(n)$	Deterministic input or simply control input; a value, in discrete time domain
$\mathbf{VEI}(n)$	Variance error information in decentralized Kalman filter; squared matrix, in discrete time domain
$\mathbf{V}(n)$	Cost-weighting (control penalty) matrix in discrete-time linear-quadratic regulator
$\mathbf{v}(n)$	Observation or sensor noise; (value in CoKF); (vector in CKF, DKF, AKF, ADKF), in discrete time domain
$W_i(n)$	Weighting factor applied in FDI system, in discrete time domain
$\mathbf{w}(n)$	Process or system noise; (value in CoKF); (vector in CKF, DKF, AKF, ADKF), in discrete time domain

X	Input fuzzy set
$\mathbf{x}(n)$	State vector; column vector, in discrete time domain
$\mathbf{x}_{-\mathrm{FDI}}(n \mid n)$	State estimate update of a system without any FDI, in discrete time domain
$Y_i(n)$	The output applied in pre-processing of FDI system, in discrete time domain
$y(n)$	Ideal noiseless observations, in discrete time domain
$y_v(n)$	Observations of a system, in discrete time domain
$y_e(n)$	Estimated observations of Kalman filter, in discrete time domain
Y	Sequence of observations
$\delta(n - \tau)$	Kronecker delta

Һ(n)	Hevisidon pulse;
	if n<0 ? Һ(n) = 0
	if n=0 ? Һ(n) = 0.5
	if n>0 ? Һ(n) = 1;

$\boldsymbol{\Gamma}(t)$	Control vector or input coupling column vector, in continuous time domain

1 INTRODUCTION

1.1 The thesis goal

This thesis provides the reader with imposed tasks to be investigated. The thesis contains a total of seven objects been introduced below:

1. To demonstrate state estimation identity of decentralized Kalman filter, DKF, and centralized Kalman filter, CKF, in an experimental simulation study assuming uniform filter conditions. What are differences?
2. To measure a mean square error, MSE, of state estimation in simulated DKF model that contains a sensor been affected by a bias with uniform filter conditions. Simulations have to be performed with total numbers of sensors such as two and five. How does the MSE depend on the total number of sensors? Tests need to be performed, namely with:
   - constant bias,
   - linearly increased bias.
3. To measure the MSE of state estimation in simulated DKF model that contains a broken sensor with uniform filter conditions. Simulations have to be performed with two and five total numbers of sensors. How does the MSE depend on the total number of sensors?
4. To measure the MSE of state estimation in DKF model that contains a sensor affected by a drift with uniform filter conditions. Simulations have to be performed by using one, two, and five total numbers of sensors. How does the MSE depend on the total number of sensors?
5. To find out a fault detection and isolation, FDI, algorithm been possibly useable for DKF model. This model has to be tested with a minor number of affected sensors by exponentially descending drifts. What is a performance of this DKF model with applied FDI algorithm, when:
   - one sensor is affected by the drift, and additionally two sensors are correct;
   - two sensors are affected by two miscellaneous drifts, plus one correct sensor?
6. To test an algorithm of adaptive Kalman filter, AKF, supported by fuzzy logic to meet the Kalman filter functionality, by definition of an optimal state estimation in Kalman filtering, with a strategy of an adapted process noise covariance $\boldsymbol{Q}$ matrix and observation R_1 noise

variance. Suppose both prior $\boldsymbol{Q}$ and R_1 are unknown. The reasonable effort should be made to additionally estimate the covariance matrices when the noise streams are unknown. So that, the AKF extends KF functionality to meet the practical needs when changing from traditional KF.

7. To demonstrate a functionality of an optimal regulator in Linear Quadratic Gaussian, LQG, problem, so that a single/multi dimensional output of a linear system takes control in backward closed loop. In addition to linear system, the (adaptive) Kalman filter should be investigated.

All solutions of given questions outlined in the Section 1.3.

1.2 The solution approach

To answer previous questions, the author has programmed a number of Kalman filter models beyond MATLAB program. The results of those simulated models will be used to answer the questions of Section 1.1 in conclusions of next units. An author's resource CD enclosed to this thesis is coming with some solutions such as MATLAB programs, Simulink models; references coupled with chapters, thesis' documentation of manuscript and its appropriate presentation file; and enough of another material. This section permits us to download the above listed data on page http://www.hut.fi/u/digana/MT.html. The author additionally provides a reader with on-line data and HTML thesis enabling every reader to link on the above web-page which can be indeed entered at any point of Chapter from one to 13. Additionally, the web-page is undertaken by the important information of this master's thesis within the author's study in 2004. It helps the reader to gather all the information about the writer's focus. Please link on page http://www.hut.fi/u/digana/Doc/MT.pdf, where the PDF form of thesis book is available free of charge.

The Helsinki University of Technology is allowed to use the data in projects, educational study, etc.

1.3 The thesis outline

In order to obtain a meaningful reading, the reminder of this thesis is organized in following two parts *Kalman-Filter Techniques* and *MATLAB Simulations and Tests* :

1. Units from 2 to 5 deal with some Kalman filter, KF, techniques such as conventional Kalman filters, CoKF, used for state estimation with one sensor; centralized, CKF, and decentralized Kalman filter, DKF, used in multi-sensor fusion; and adaptive Kalman filters, AKF, used for state estimation with unknown process and sensor noise covariance matrices being adapted. The Unit 5 summarizes the proposed technique of AKF designed for CoKF, CKF and DKF topologies. In order to use a fuzzy logic approach in an adaptive process, more detailed exercises of this AKF called adaptive fuzzy logic KF is explored in Unit 13. As we can see, this thesis devotes to KF which comprise a number of methods, types and topologies depending on use and computing complexity of applied processors.
2. Units from 6 to 14 cover some KF tests, equation implementations, a KF functionality and reliability of different structures of KF.

Finally, main summary and conclusions of this work are given in Unit 15.

"*Make yourself comfortable in a chair before each reading.*"

This section is related to questions of Section 1.1 addressing all answers to thesis. Their corresponding answers are organized as follows:

1. **Solution of the first question.** We will view a mathematical identity of CKF and DKF presented by some experimental results. We suppose that all signals are broadcasted without any unknown latency or transmission failure in the models of multi-sensor fusion. To answer the first question we will follow the consecution:
    - In Unit 6, a state estimation accuracy of conventional Kalman filter, CoKF, is measured to get around a possible model imperfection. The Model 2 of CoKF disposes of a complex inverse covariance matrix to be computed;

- In Unit 7, the CKF models are tested toward their correctness. Also there a computational complexity of inverse covariance matrix would be solved;
- Finally, the performance of CKF and DKF models can be compared by measured MSE and state estimation accuracy. A difference between an estimated state vector of CKF and DKF is measured in Unit 8.

2. **Solution of the second question.** In Unit 9, four simulations are presented with two and five sensors in DKF model, and optionally with two levels of power of a process noise. We will clarify a relation of MSE in the DKF model and contained total number of sensors such two and five. Typically, every fault of a sensor increases MSE of state estimation and decreases state estimation accuracy.
3. **Solution of the third question.** In Unit 10, results of two simulations are shown to find a relation of MSE values in DKF model and two or five sensors, where only one sensor is broken. The MSE formulation is used the same as previously with the same two options of power of process noise. A broken sensor produces zero-signal received by DKF.
4. **Solution of the fourth question.** In Unit 11, three simulations are performed using DKF model with one corrupted sensor by exponentially descending drift. The point is based on comparison of three MSE values between the models, which have optionally one, two and five sensors. Additionally, two power levels of process noise are considered.
5. **Solution of the fifth question.** In Unit 12, fault detection and isolation, FDI, algorithm is presented. Sections 12.1 and 12.2 refer to results of DKF model containing one and two sensors affected by exponentially descending drifts, respectively. Also two options of high and low power of process noise are considered in each section. The main idea is covered by comparison of FDI reliability between DKF models with FDI system inside of each node. There is zero FDI reliability in DKF model without any FDI system. The reliability tends to power compensation performed by FDI of state estimates to reach high state estimation accuracy in DKF proposed model.
6. **Solution of the sixth question.** Problem of adaptive fuzzy logic Kalman filter is shared further in two parts. The first part deals with an algorithm of adaptive schema presented in Unit 13, and the second part deals with some exercises in adaptive fuzzy logic Kalman filter shown further in Section 13.1 and 13.2. The Sections 13.1 and 13.2 present some tests of third and tenth order filter, respectively. The reason to use two filters of the same quality

despite different dimension of state vectors focuses on state estimation accuracy measured enquiring a sensitivity of uncertain parameters $\boldsymbol{Q}(n)$ and $\boldsymbol{R}(n)$ in an example of adaptive system. Additionally, every test is a subject of two simulations considering high and low power of process noise.

7. **Solution of the seventh question.** First of all, linear-quadratic regulator, LQR, problem is solved in Unit 14. There two systems are used such as third and tenth order filter. The systems are considered with high and low power of process noise to be used in MATLAB simulations. Solving the LQR problem, state-feedback LQR gain and feed-forward gain are obtained and used in simulations of next units. Section 14.2 focuses on linear-quadratic Gaussian, LQG, problem based regulator in noisy environment containing one sensor. Next Section 14.3 deals with the same problem in system of multi sensors where an adaptive decentralized Kalman filter, ADKF, and LQR are used according to separation principle. Three independent sensors are taken too. An exceptional simulation occurs in Section 14.1 where non-linear actuator affects a control process in solution of LQR problem.

2 KALMAN FILTER TECHNIQUES

In this unit, we dedicate the effort to introduce Kalman filter - KF techniques with three models of conventional Kalman filter, CoKF, mainly. Although there is no difference between centralized Kalman filter CKF and CoKF, we like to show the CoKF as an estimator structure in single-sensor systems, [8]. We are not interested in equation derivation but we just present KF as the optimal estimator. The interested reader can read more on accurate KF equation derivation in [1] - [4].

First of all, we will assume a mathematical model of a plant defined by equations of discrete system dynamics. To get the equations of the optimum estimator, i.e., the KF, suppose that the plant of system dynamics are designed by the (possibly time-varying) general model of linear finite-dimensional stochastic system, see below; [1], [2].

$$\boldsymbol{x}(n+1)=\boldsymbol{A}\boldsymbol{x}(n)+\boldsymbol{B}w(n) \qquad (2\text{-}1)$$

A control input $u(n)$ of plant is included in process noise $w(n)$, where n refers to discrete time. The $w(n)$ noise is not necessary white but it should be a zero mean noise.

$$y_v(n)=\boldsymbol{C}\boldsymbol{x}(n)+v(n), \qquad n \geq n_0 \qquad (2\text{-}2)$$

The $n_0 = 0$ is the initial time. The equation (2-1) is called stochastic state transition or system model equation, and (2-2) is called the observation equation of stochastic model, [9]. In the terminology, $\boldsymbol{A}$ is state transition matrix, $\boldsymbol{B}$ is input coupling vector, $\boldsymbol{C}$ is observation vector, $\boldsymbol{x}$ is called state vector and $y_v(n)$ is plant observation and finally $v(n)$ is called sensor or observation noise. The $\boldsymbol{x}(n_0)$ has a mean $\boldsymbol{x_0}$ and state covariance $\boldsymbol{P_0}$ matrix and

$$\left\{\begin{bmatrix}\boldsymbol{v}(n)\\ \boldsymbol{w}(n)\end{bmatrix}\begin{bmatrix}\boldsymbol{v}(\tau)^{\mathrm{T}} & \boldsymbol{w}(\tau)^{\mathrm{T}}\end{bmatrix}\right\}=\begin{bmatrix}\boldsymbol{Q} & \boldsymbol{S}\\ \boldsymbol{S}^{\boldsymbol{T}} & \boldsymbol{R}\end{bmatrix}\delta(n-\tau); \qquad (2\text{-}3)$$
$$\boldsymbol{Q} \geq \boldsymbol{0}, \boldsymbol{R} \geq \boldsymbol{0}$$

where $$\boldsymbol{w} \sim N(0,\boldsymbol{Q}(n)), \qquad (2\text{-}4)$$

and $$\boldsymbol{v} \sim N(0,\boldsymbol{R}(n)), \text{[1].} \qquad (2\text{-}5)$$

Here we denote equation (2-4) for (2-1), and (2-5) for (2-2). The $\boldsymbol{Q}$ is called process noise covariance matrix and $\boldsymbol{R}$ is observation noise covariance matrix.

The relation of $\boldsymbol{w}(n)$, $\boldsymbol{v}(n)$ and $\boldsymbol{Q}(n)$, $\boldsymbol{R}(n)$ is respectively defined by (2-4), (2-5). Also we assume the ideal non-noisy observations

$$y(n) = \boldsymbol{Cx}(n). \tag{2-6}$$

The discrete plant model was already described by the applied process noise $w(n)$. Observations are given by equation (2-2). Although the observation noise $v(n)$ is generated by sensor but not by a plant. The equations (2-1) and (2-2) both describe state-space model. Below we mention some conditions been valid for a KF such as an optimal state estimator and its equations, [1] and [2].

1. The sampled white noise has a mean of zero:

$$E[\boldsymbol{w}(n)] = 0; \qquad E[\boldsymbol{v}(n)] = 0.$$

2. The $\boldsymbol{w}(n)$ and $\boldsymbol{v}(n)$ are uncorrelated for $n \neq k$, i.e.:

$\boldsymbol{S} = E[\boldsymbol{v}(n) \ \ \boldsymbol{w}^T(k)] = \boldsymbol{0}, \qquad for \ \ n \neq k$, where k refers to a discrete time.

3. Noise variances are

$$E[\boldsymbol{w}(n) \ \ \boldsymbol{w}^T(k)] = \begin{cases} \boldsymbol{Q}(n), & for \ \ n = k \\ \boldsymbol{0}, & for \ \ n \neq k \end{cases}$$

$$E[\boldsymbol{v}(n) \ \ \boldsymbol{v}^T(k)] = \begin{cases} \boldsymbol{R}(n), & for \ \ n = k \\ \boldsymbol{0}, & for \ \ n \neq k \end{cases}.$$

4. $E\{[\boldsymbol{x}(n) - \boldsymbol{x}(n \mid n)]\boldsymbol{v}^T(n)\} = \boldsymbol{0}$, where $\boldsymbol{x}(n)$ is state vector of plant, $\boldsymbol{x}(n \mid n)$ is estimated state.

5. State error covariance matrix $\boldsymbol{P}(n)$, innovation Kalman filter gain $\boldsymbol{M}(n)$, $\boldsymbol{A}$, $\boldsymbol{C}$, $\boldsymbol{R}$, $\boldsymbol{Q}$ and $\boldsymbol{S} = \boldsymbol{0}$ are independent of observations sequence $\boldsymbol{Y}(n-1) \cong \{y_v(n-1), y_v(n-1), \ldots, y_v(0)\}$.

6. We shall specify new symbol, namely $r(n)$ to the error of $y_v(n)$-$\boldsymbol{Cx}(n|n-1)$ the innovations sequence (residuals), where $E\{r(n) \mid \boldsymbol{Y}(n-1)\} = 0$ and $\boldsymbol{x}(n|n-1)$ is the state time update in Kalman filter.

Nonzero mean of a noise is not our case of study and tests. In Kalman filtering we consider $\boldsymbol{Q}(n) < \boldsymbol{x}(n)\boldsymbol{x}^T(n)$ and $\boldsymbol{R}(n) < y_v(n){y_v}^T(n)$. We assume that the system output can be predicated and a white noise of innovations sequence-residuals $r(n)$, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are correctly estimated. Unfortunately, this is not the reality. Supposing those assumptions, the optimality of state estimation is achieved when an algebraic constraint on derived equations is used in *Model 1, 2* and *3 of Kalman filter* below. Solving the estimation problem, KF minimizes the state error covariance matrix in optimal linear filtering. The innovations sequence named by $r(n)$ is useful for the reason of state estimation in Kalman filtering.

Model 1 of Kalman filter. In this part, the author investigates a timing diagram of KF in order to get a control program flow with applied equations in Table 2-1 below. This will be also introduced briefly in next model. The model refers to [1 - 3]. Appendix A.2 shows MATLAB model that is organised as a two-stage filter, where the first stage of state estimation is shown in blue and the second stage in red. The table deals with two programs, i.e. initial program and main iterative program. The initial time n_0 is the formal time when processor does not process first sample but starts an initial program.

	Initial Program **Initial Time n = 0**	
Initial error covariance:	$\mathbf{P}(n\|n-1) = \mathbf{B}\,\mathbf{Q_d}\,\mathbf{B}^T$, where $\mathbf{Q_d}$ is defaulted $\mathbf{Q}(0) > \mathbf{0}$	(2-7)
Initial condition on the state:	$\mathbf{x}(n\|n-1) = \mathbf{0}$	(2-8)
Initial condition on Filter Output:	$y_e(n) = 0$	(2-9)
	Iteration Time n = 1,2,3,...	
Observation update:		
Innovation KF Gain:	$\mathbf{M}(n) = \mathbf{P}(n\|n-1)\,\mathbf{C}^T / (\mathbf{C}\,\mathbf{P}(n\|n-1)\,\mathbf{C}^T + R(n))$	(2-10)
Innovations Sequence (Residuals):	$r(n) = y_v(n) - \mathbf{C}\,\mathbf{x}(n\|n-1)$	(2-11)
State Estimate Update:	$\mathbf{x}(n\|n) = \mathbf{x}(n\|n-1) + \mathbf{M}(n)\,r(n)$	(2-12)
Error Covariance Update:	$\mathbf{P}(n\|n) = [\mathbf{I} - \mathbf{M}(n)\,\mathbf{C}]\,\mathbf{P}(n\|n-1)$	(2-13)
Estimated Filter Output:	$y_e(n) = \mathbf{C}\,\mathbf{x}(n\|n)$	(2-14)
Error Covariance:	$\text{error cov} = \mathbf{C}\,\mathbf{P}(n\|n)\,\mathbf{C}^T$	(2-15)
Time update:		
State Time Update:	$\mathbf{x}(n+1\|n) = \mathbf{A}\,\mathbf{x}(n\|n) + \mathbf{B}\,u(n)$	(2-16)
Error Covariance Time Update:	$\mathbf{P}(n+1\|n) = \mathbf{A}\,\mathbf{P}(n\|n)\,\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}(n)\,\mathbf{B}^T$	(2-17)

Table 2-1 Model 1 of KF.

Every KF works by computing residuals with (2-11) formula and consequently estimating a state vector using (2-12). Additionally, our objective tends to find the appropriate value of innovation KF gain $\boldsymbol{M}(n)$ for the optimal filtering. Roughly spoken, the Kalman filter is nothing, but the optimal linear estimator describing set of equations (for instance in Table 2-1) used to estimate the state vector with the knowledge of state-space model, see (2-1) and (2-2).

The timing diagram of Table 2.1 can be described as follows:

1. The innovation KF gain is computed in (2-10) with the usage of delayed matrix of error covariance time update, (2-17), and the recent observation noise variance ***R***(*n*) at the beginning of discrete time *n*. This refers to computation of state error covariance matrix, which indicates an accuracy of the state estimate. This calculation provides optimal innovation KF gain to minimize a KF cost function below:
$$\boldsymbol{P}(n) = E\left([\boldsymbol{x}(n) - \boldsymbol{x}(n \mid n)][\boldsymbol{x}(n) - \boldsymbol{x}(n \mid n)]^T \right), \tag{2-18}$$
regarding to the notes of KF properties on page 7, thus the KF is an optimal estimator. From KF theory, [2], the time update of state error covariance matrix can be obtained as follows:
$$\boldsymbol{P}(n+1 \mid n) = \boldsymbol{A_S}(n)\boldsymbol{P}(n \mid n)\boldsymbol{A_S}(n)^T + \boldsymbol{BQ}(n)B^T + \boldsymbol{M}(n)\boldsymbol{R}(n)\boldsymbol{M}(n)^T \,, \tag{2-19}$$
where steady-state matrix $\boldsymbol{A_S}(n) = \boldsymbol{A}(\boldsymbol{I} - \boldsymbol{M}(n)\boldsymbol{C})$. Hence the aim tends to find ***M***(*n*) that minimizes (2-19) as we did to satisfy the minimization of (2-18) in optimal filtering. This is accomplished by the principle of optimality, assuming that (2-18) was already optimised by the choice of ***M***(*n-1*), ***M***(*n-2*), etc. If we could, we would like to differentiate the right hand side of (2-19) with respect to ***M***(*n*) and set the result to zero. Recently, we have found the optimum gain (2-10) that minimised (2-19) with the given ***Q*** and ***R*** in Kalman filtering. After substitution of (2-10) into (2-19) the formulas (2-13) and (2-17) are found, see [2].
2. The innovations sequence (2-11) is computed in the second step, where ***x***(*n*|*n-1*) was obtained by one step-ahead predictor of state time update (2-16) at previous discrete time *n-1*. The innovations sequence (2-11) is the difference between the observation $y_v(n)$ of plant (2-2) and the priori observation ***Cx***(*n*|*n-1*).
3. The state estimate update (2-12) can be computed.
4. Already mentioned error covariance update (2-13) was computed at *n-1*. Now, the estimated filtered output (2-14) is given at *n*.
5. Finally, equations of state time update (2-16) and error covariance time update (2-17) are computed and stored into a memory to be ready for the processing of new iteration at time *n+1*.

There are more alternatives, but for example (2-15) may not be necessary computed here. It measures priori error covariance seen at the output of KF. The essence of Table 2-1 is that this timing diagram is correct for extended Kalman filter, EKF, where $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ matrices depend on $\boldsymbol{x}(n)$ and for a time-varying process where the time order of equations is required. Of course, we implement this model into centralized Kalman filter, CKF, and a local filter of decentralized Kalman filter DKF.

Model 2 of Kalman filter. A timing diagram shown in Table 2-2 is modified model from Table 2-1. Both models are mathematically identical. A difference between these two models and their mathematical identity is measured because other expressions of error covariance update and innovation gain are used here. This way, we will slightly tend to DKF techniques in multi sensor fusion.

Initial Program		
Initial Time n = 0		
Initial error covariance:	$\mathbf{P}(n\|n-1) = \mathbf{B}\,\mathbf{Q_d}\,\mathbf{B}^T$, where $\mathbf{Q_d}$ is defaulted $\mathbf{Q}(0) > \mathbf{0}$	(2-20)
Initial condition on the state:	$\mathbf{x}(n\|n-1) = \mathbf{0}$	(2-21)
Initial condition on Filter Output:	$y_e(n) = 0$	(2-22)
Iteration Time n = 1,2,3,…		
Observation update :		
Error Covariance Update:	$\mathbf{P}(n\|n)^{-1} = \mathbf{P}(n\|n-1)^{-1} + \mathbf{C}^T\,\mathbf{C} / R(n)$ $\mathbf{P}(n\|n) = [\mathbf{P}(n\|n)^{-1}]^{-1}$	(2-23)
Innovation KF Gain:	$\mathbf{M}(n) = \mathbf{P}(n\|n)\,\mathbf{C}^T / R(n)$	(2-24)
Innovations Sequence (Residuals):	$r(n) = y_v(n) - \mathbf{C}\,\mathbf{x}(n\|n-1)$	(2-25)
State Estimate Update:	$\mathbf{x}(n\|n) = \mathbf{x}(n\|n-1) + \mathbf{M}(n)\,r(n)$	(2-26)
Estimated Filter Output:	$y_e(n) = \mathbf{C}\,\mathbf{x}(n\|n)$	(2-27)
Error Covariance:	$\text{error cov} = \mathbf{C}\,\mathbf{P}(n\|n)\,\mathbf{C}^T$	(2-28)
Time update :		
State Time Update:	$\mathbf{x}(n+1\|n) = \mathbf{A}\,\mathbf{x}(n\|n) + \mathbf{B}\,u(n)$	(2-29)
Error Covariance Time Update:	$\mathbf{P}(n+1\|n) = \mathbf{A}\,\mathbf{P}(n\|n)\,\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}(n)\,\mathbf{B}^T$	(2-30)

Table 2-2 Model 2 of KF.

In this model, the (2-23) is used instead of (2-13) with (2-24). Both (2-23) and (2-24) modify this model. With regard to the Section 1.3, we will verify mathematical identity of (2-23) toward (2-13) in Unit 6. Besides, the (2-24) can not be used instead of (2-10) in Table 2-1, however both are mathematically identical, otherwise an algebraic loop causes. Appendix A.3 shows two-stage filter in this model in MATLAB program.

Model 3 of Kalman filter. This model is presented in Table 2-3 and Appendix A.4.

Initial Program		
Initial Time n = 0		
Initial error covariance:	$\mathbf{P}(n \mid n\text{-}1) = \mathbf{B}\,\mathbf{Q_d}\,\mathbf{B}^T$, where $\mathbf{Q_d}$ is defaulted $\mathbf{Q}(0) > \mathbf{0}$	(2-31)
Initial condition on the state:	$\mathbf{x}(n \mid n\text{-}1) = \mathbf{0}$	(2-32)
Initial condition on Filter Output:	$y_e(n) = 0$	(2-33)
Iteration Time n = 1,2,3,...		
Observation update :		
Innovation KF Gain:	$\mathbf{M}(n) = \mathbf{P}(n \mid n\text{-}1)\,\mathbf{C}^T / [\mathbf{C}\,\mathbf{P}(n \mid n\text{-}1)\,\mathbf{C}^T + R(n)]$	(2-34)
Steady-State Filter State Transition Matrix:	$\mathbf{A_S} = \mathbf{A}[\mathbf{I} - \mathbf{M}(n)\mathbf{C}]$	(2-35)
Steady-State Filter Observation Vector:	$\mathbf{C_S} = \mathbf{C}[\mathbf{I} - \mathbf{M}(n)\,\mathbf{C}]$, without R factorization	(2-36)
Error Covariance Update:	$\mathbf{P}(n \mid n) = [\mathbf{I} - \mathbf{M}(n)\mathbf{C}]\,\mathbf{P}(n \mid n\text{-}1)$	(2-37)
Estimated Filter Output:	$y_e(n) = \mathbf{C_S}\,\mathbf{x}(n \mid n\text{-}1) + \mathbf{C}\,\mathbf{M}(n)\,y_v(n)$	(2-38)
Error Covariance:	error cov = $\mathbf{C}\,\mathbf{P}(n \mid n)\,\mathbf{C}^T$	(2-39)
Time update :		
State Time Update:	$\mathbf{x}(n+1 \mid n) = \mathbf{A_S}\,\mathbf{x}(n \mid n\text{-}1) + \mathbf{B}\,u(n) + \mathbf{A}\,\mathbf{M}(n)\,y_v(n)$	(2-40)
Error Covariance Time Update:	$\mathbf{P}(n+1 \mid n) = \mathbf{A}\,\mathbf{P}(n \mid n)\,\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}(n)\,\mathbf{B}^T$	(2-41)

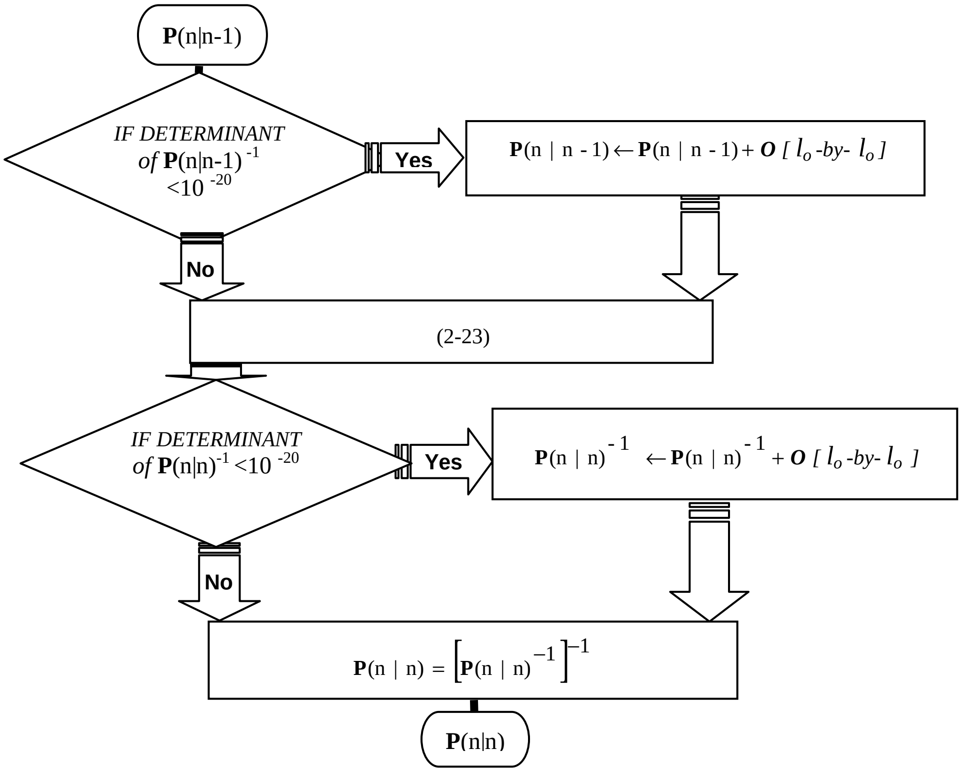
Table 2-3 Model 3 of KF.

This model of KF exempts state estimate update that already is mathematically included in (2-41). The estimated observation of KF, $y_e(n)$, additionally called filtered output is computed directly from state time update $\boldsymbol{x}(n|n\text{-}1)$, see (2-38). This model can be applied for output filtering $y_e(n)$ and state prediction $\boldsymbol{x}(n|n\text{-}1)$, or state estimation $\boldsymbol{x}(n|n)$ should be additionally extracted if desired. All models are mathematically specified in compliance with references [1] - [4], and [7] - [10]. The structure of KF as the two-stage filter is designed according to reference [14].

2.1 Computational improvement

We just went through the Kalman filter techniques, where some difficulties belong into simulations made on computer. Therefore this Section, 3.1 and 4.1 will be proceeded to overcome some problems, which are couplet with discrete and digital estimators.

The state error covariance matrix becomes singular always at the beginning of simulation. We present following method capable to avoid singularity problem during the term of simulation. Elements $\boldsymbol{o}_{\mathrm{ii}}$ refer to $N(0{,}10^{-14})$ and identity matrix $\boldsymbol{O}$ of noise $N(0{,}10^{-14})$. Those elements $\boldsymbol{o}_{\mathrm{ii}}$ are taken to absolute value $\boldsymbol{O}=\left[\boldsymbol{O}_{\mathrm{ii}}\right]=\left|\boldsymbol{o}_{\mathrm{ii}}\right|$, where $i=1,\ \ 2,\ \ 3,\ \ ...,$ l_{o}, and l_o is the order of state vector. The matrix is additionally summarised with state error covariance matrix when the singularity happens. This procedure of improvement presented by Figure 2.1-1 is shown in Appendix A.5. In the same way, the inverse covariance matrix is processed. The Figure 2.1-1 refers to problem of (2-23). In the flowchart, the decision blocks indicate error conditions of the covariance and inverse covariance matrices when numerically getting near the singularity. Then the matrices need to be adjusted using the above method when a determinant of $\boldsymbol{P}(n|n)$ or $\boldsymbol{P}(n|n)^{-1}$ is actually below 10^{-20} of threshold. This reasonable improvement describes benefit in simulations computing.

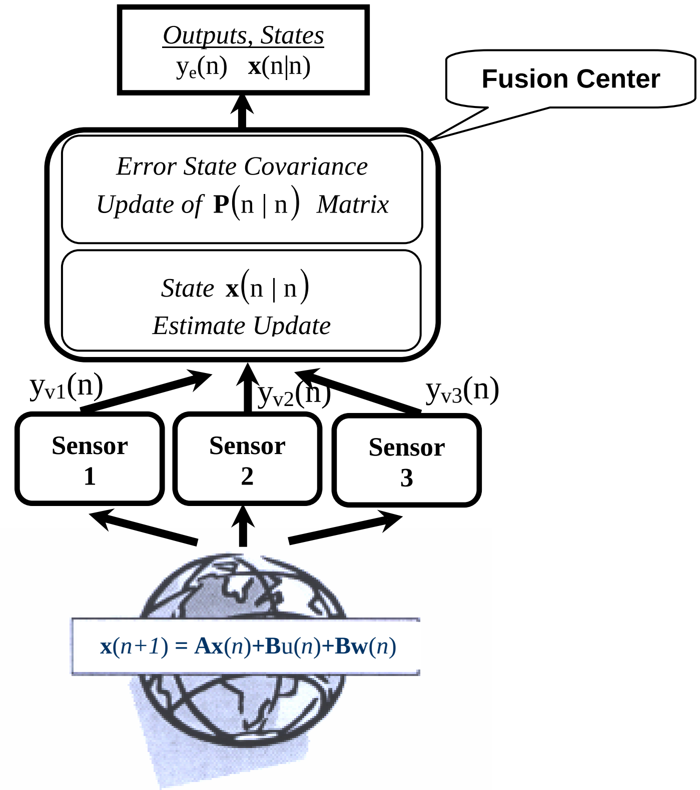


P(n|n-1)
IF DETERMINANT of P(n|n-1)^-1 <10^-20
Yes
P(n | n - 1) ← P(n | n - 1) + O [l_o-by- l_o]
No
(2-23)
IF DETERMINANT of P(n|n)^-1 <10^-20
Yes
P(n | n)^-1 ← P(n | n)^-1 + O [l_o-by- l_o]
No
P(n | n) = [P(n | n)^-1]^-1
P(n|n)

Figure 2.1-1 Treatment of the inverse covariance matrix computation.

3 CENTRALIZED KALMAN FILTER TECHNIQUES

This unit deals with CKF technique and models. The models are built according to [15]. In centralized Kalman filtering, signals of sensors are transferred through the communication network to the central processor to generate the optimal central estimate $\boldsymbol{x}(n|n)$. The all information is sent to the fusion centre, Figure 3-1, to yield $\boldsymbol{x}(n|n)$ and minimize state estimation error.



Outputs, States
$y_e(n)$ $\mathbf{x}(n|n)$
Fusion Center
Error State Covariance Update of $\mathbf{P}(n \mid n)$ Matrix
State $\mathbf{x}(n \mid n)$ Estimate Update
$y_{v1}(n)$
$y_{v2}(n)$
$y_{v3}(n)$
Sensor 1
Sensor 2
Sensor 3
$\mathbf{x}(n+1) = \mathbf{Ax}(n)+\mathbf{B}u(n)+\mathbf{Bw}(n)$

Figure 3-1 Centralized Kalman filter topology.

All sensors are measuring outputs of plant. Of course, all the plant is controlled by one actuator which gives $u(n)$ value the same for CKF estimator input. Three sensors presented in Figure 3-1 produce observations $y_{v1}(n)$, $y_{v2}(n)$ and $y_{v3}(n)$ which are necessarily not the same. The observations are sent from each sensor to fusion centre that performs central state estimation, see Figure 3-1.

For example the $y_{v1}(n)$ may represent the first CCD camera observation, $y_{v2}(n)$ represents the microphone observation and $y_{v3}(n)$ represents the mechanical sensor of a track and position estimation of a train in a tunnel, [16]. Observations $y_{vi}(n)$ are modeled as filtered $\boldsymbol{x}(n)$ through observation vectors $\boldsymbol{C}_i$, where $\boldsymbol{x}(n)$ is only one. Subscript $i = 1, 2, 3$ refers to the particular sensors.

Model 1 of centralized Kalman filter. A timing diagram of this model is shown in Table 3-1 as a special case of *Model 2 of Kalman filter* of Table 2-1 with sensor fusion. In other words, signals are combined to get state estimation in fusion. Initial program starts at n_0 and prepares (3-1) – (3-3).

Initial Program **Initial Time n = 0**		
Initial error covariance:	$\mathbf{P}(n \mid n-1) = \mathbf{B\,Qd\,B}^T$, $\mathbf{Q_d}$ is defaulted $\mathbf{Q}(0) > \mathbf{0}$	(3-1)
Initial condition on the state:	$\mathbf{x}(n \mid n-1) = \mathbf{0}$	(3-2)
Initial condition on Filter Output:	$y_{ei}(n) = 0$	(3-3)
Iteration Time n = 1,2,3,...		
Observation update :		
Innovation KF Gain:	$\mathbf{M}_i(n) = \mathbf{P}(n \mid n-1)\,\mathbf{C}_i^T \Big/ \left[\mathbf{C}_i\,\mathbf{P}(n \mid n-1)\,\mathbf{C}_i^T + R_i(n)\right]$	(3-4)
Innovations Sequence (Residuals):	$r_i(n) = y_{vi}(n) - \mathbf{C}_i\,\mathbf{x}(n \mid n-1)$	(3-5)
State Estimate Update:	$\mathbf{x}(n \mid n) = \mathbf{x}(n \mid n-1) + \frac{1}{N_o} \sum_{i=1}^{N_o} \mathbf{M}_i(n)\,r_i(n)$, $N_o \geq 1$ N_o means number of sensors	(3-6)
Error Covariance Update:	$\mathbf{P}(n \mid n) = \left[\mathbf{I} - \frac{1}{N_o} \sum_{i=1}^{N_o} \mathbf{M}_i(n)\,\mathbf{C}_i\right] \mathbf{P}(n \mid n-1)$	(3-7)
Estimated Filter Output:	$y_{ei}(n) = \mathbf{C}_i\,\mathbf{x}(n \mid n)$	(3-8)
Error Covariance:	$\text{error cov}_i = \mathbf{C}_i\,\mathbf{P}(n \mid n)\,\mathbf{C}_i^T$	(3-9)
Time update :		
State Time Update:	$\mathbf{x}(n+1 \mid n) = \mathbf{A}\,\mathbf{x}(n \mid n) + \mathbf{B}\,u(n)$	(3-10)
Error Covariance Time Update:	$\mathbf{P}(n+1 \mid n) = \mathbf{A}\,\mathbf{P}(n \mid n)\,\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}(n)\,\mathbf{B}^T$	(3-11)

Table 3-1 Timing diagram of Model 1 of CKF.

State estimation is performed by (3-6), and fusion is shown in the sum. Estimation of error state covariance matrix is in (3-7) with fusion of innovation gain matrix $\boldsymbol{M}_i(n)$. This fusion works without any signal selection, but the averaging sum is only used. The information $y_{vi}(n)$ over all sensors is applied to calculation of mean information to get one value such as only one column state vector, and only one squared covariance matrix.

The state estimation of $\boldsymbol{x}(n|n)$ is updated by $\boldsymbol{M}_i(n)r_i(n)$ in the sum of (3-6), where $r_i(n)$ is the function of sensor observations $y_{vi}(n)$ in (3-5). Error covariance is updated by $\boldsymbol{M}_i(n)\boldsymbol{C}_i$ in the sum of (3-7). There the innovation gain matrix depends on observation (sensor) noise variance $R_i(n)$ of the i-th sensor in (3-4). The error covariance time update depends on $\boldsymbol{Q}(n)$ of stochastic model of plant in (3-11). The observation vectors can be at time-relation functions of $\boldsymbol{C}_i(n)$ in time-varying models.

Model 2 of centralized Kalman filter with inverse covariance matrix. In this model of Table 3-2 we assume same initial program as well as the Table 3-1.

Iteration Time n = 1,2,3,...		
Observation update :		
Error Covariance Update:	$\mathbf{P}(n \mid n)^{-1} = \mathbf{P}(n \mid n-1)^{-1} + \frac{1}{N_o} \sum_{i=1}^{N_o} \mathbf{C}_i^T \mathbf{C}_i / R_i(n)$ $\mathbf{P}(n \mid n) = \left[\mathbf{P}(n \mid n)^{-1}\right]^{-1}$	(3-12)
Innovation KF Gain:	$\mathbf{M}_i(n) = \mathbf{P}(n \mid n)\ \mathbf{C}_i^T / R_i(n)$	(3-13)
Innovations Sequence (Residuals):	$r_i(n) = y_{vi}(n) - \mathbf{C}_i \mathbf{x}(n \mid n-1)$	(3-14)
State Estimate Update:	$\mathbf{x}(n \mid n) = \mathbf{x}(n \mid n-1) + \frac{1}{N_o} \sum_{i=1}^{N_o} \mathbf{M}_i(n)\ r_i(n),\ N_o \geq 1$ N_o means number of sensors	(3-15)
Estimated Filter Output:	$y_{ei}(n) = \mathbf{C}_i\ \mathbf{x}(n \mid n)$	(3-16)
Error Covariance:	$\text{error cov}_i = \mathbf{C}_i\ \mathbf{P}(n \mid n)\, \mathbf{C}_i^T$	(3-17)
Time update :		
State Time Update:	$\mathbf{x}(n+1 \mid n) = \mathbf{A}\, \mathbf{x}(n \mid n) + \mathbf{B}\, u(n)$	(3-18)
Error Covariance Time Update:	$\mathbf{P}(n+1 \mid n) = \mathbf{A}\, \mathbf{P}(n \mid n)\ \mathbf{A}^T + \mathbf{B}\, \mathbf{Q}(n)\, \mathbf{B}^T$	(3-19)

Table 3-2 Timing diagram of Model 2 of CKF.

The *Model 2 of centralized Kalman filter*, Table 3-2, is a special instance of *Model 1 of CKF*, Table 3-1. Hence (3-12) is used instead of (3-7).

Model 3 of fully centralized Kalman filter. In this model of Table 3-3 also we assume the same initial program as in the previous one. The built MATLABs model can be seen in Appendix B.1.

Iteration Time n = 1,2,3,...

Observation update :

Error Covariance Update:

$$\mathbf{P}(n \mid n)^{-1} = \mathbf{P}(n \mid n-1)^{-1} + \frac{1}{N_o} \sum_{i=1}^{N_o} \mathbf{C}_i^T \mathbf{C}_i / R_i(n) \qquad (3\text{-}20)$$

$$\mathbf{P}(n \mid n) = \left[\mathbf{P}(n \mid n)^{-1}\right]^{-1}$$

State Estimate Update:

$$\mathbf{x}(n \mid n) = \mathbf{P}(n \mid n) \left[\mathbf{P}(n \mid n-1)^{-1} \mathbf{x}(n \mid n-1) + \frac{1}{N_o} \sum_{i=1}^{N_o} \mathbf{C}_i^T / R_i \, y_{vi}(n) \right] \qquad (3\text{-}21)$$

Estimated Filter Output:

$$y_{ei}(n) = \mathbf{C}_i \, \mathbf{x}(n \mid n) \qquad (3\text{-}22)$$

Error Covariance:

$$\text{error cov}_i = \mathbf{C}_i \, \mathbf{P}(n \mid n) \, \mathbf{C}_i^T \qquad (3\text{-}23)$$

Time update :

State Time Update:

$$\mathbf{x}(n+1 \mid n) = \mathbf{A}\,\mathbf{x}(n \mid n) + \mathbf{B}\,u(n) \qquad (3\text{-}24)$$

Error Covariance Time Update:

$$\mathbf{P}(n+1 \mid n) = \mathbf{A}\,\mathbf{P}(n \mid n)\,\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}(n)\,\mathbf{B}^T \qquad (3\text{-}25)$$

Table 3-3 Timing diagram of Model 3 of CKF.

Those equations of Table 3-3 are derived from Table 3-1 and 3-2 with regard to reference material [10, 15] issues. Sensed observations $y_{vi}(n)$ are directly applied with fusion to state estimation of $\boldsymbol{x}(n|n)$ in (3-21). An innovation gain is mathematically eliminated here. The error covariance, (3-20), is updated in a sum with observation (sensor) noise variance $R_i(n)$. This procedure of information collection can be seen as a fusion in this estimator. In all CKF models, the $R_i(n)$ and $\boldsymbol{Q}(n)$ are needed to be already known at every discrete time at the point of sensors and $u(n)$ as a property of Kalman filter in Unit 2.

The concept of CKF technique is analysed in papers [10], [14] and [15].

3.1 Computational improvement

Time updated covariance matrix becomes singular always at beginning of simulation and in a term of middle time of simulation. This inconvenience causes wrong state estimate update in digital computing. To get over this problem we present following method capable to avoid the singularity during whole simulation shown by a flowchart in Figure 3.1-1.

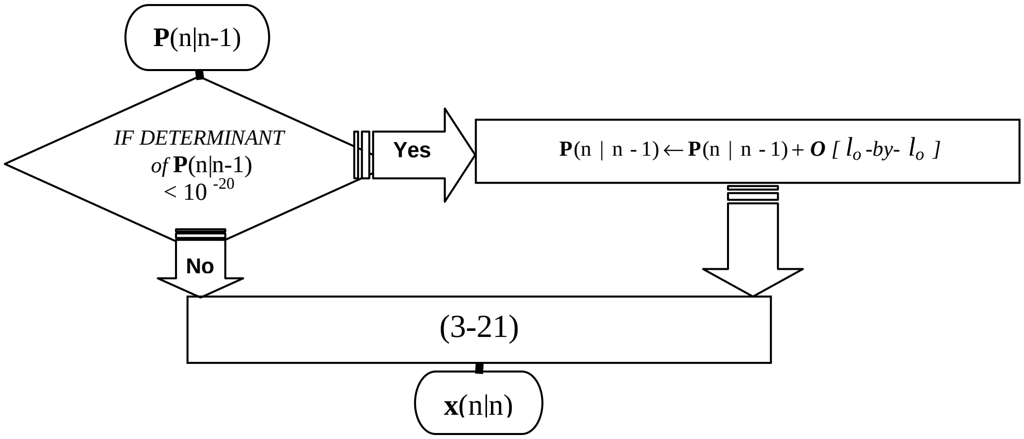

P(n|n-1)
IF DETERMINANT of P(n|n-1) < 10 -20
Yes
P(n | n - 1) ← P(n | n - 1) + O [l_o-by- l_o]
No
(3-21)
x(n|n)

Figure 3.1-1 The second treatment of fully centralized KF.

For the recently introduced method, we apply elements $\boldsymbol{o}_{ii}$ that refer to $N(0{,}10^{-14})$ and identity matrix $\boldsymbol{O}$ of noise $N(0{,}10^{-14})$. Those elements $\boldsymbol{o}_{ii}$ are taken to absolute value $\boldsymbol{O}=[\boldsymbol{O}_{ii}]=|\boldsymbol{o}_{ii}|$, where $i=1,\ 2,\ 3,\ ...,\ \mathrm{l_o}$, and l_o is the order of state vector. This improvement works by summarising $\boldsymbol{O}$ matrix with the error covariance time update if the singularity occurs as described by the decision making in the flowchart. The MATLAB block model of this improvement is presented in Appendix B.2. The problem of whole covariance matrix calculation is solved by Section 2.1. Here the Figure 2.1-1 may be used in CKF models to overcome the singularity problem caused in (3-12) and (3-20). We figure out a little power of noise $\boldsymbol{o}_{ii}$ with unaffected influence on Kalman filtering.

4 DECENTRALIZED KALMAN FILTER TECHNIQUE

This unit deals with Decentralized Kalman filter - DKF technique. The decentralized Kalman filter processes data from many sensors to provide a global state estimation in multi-sensor fusion. A DKF model was built according to references [9], [11] - [13].

Every DKF contains a local and a global filter that emphasises double-estimation in a node. The local filter uses its own data and observation $y_V(n)$ to perform an optimal local estimates $\boldsymbol{P}(n|n)$ and $\boldsymbol{x}(n|n)$. These estimates are obtained in a parallel processing mode implicitly. Thus, each node takes observation (possibly asynchronously) from a plant of an environment. With this observation (and its associated variance) the DKF is able to compute a partial state estimate. Then each node broadcasts one vector and a matrix of error information to the others and it receives other information being broadcasted to it. Those two (one vector and a matrix) as state error information ***SEI***(n) and variance error information ***VEI***(n) are used for global state and covariance estimate in global filter of every node. Finally, all nodes performed same global state estimates $\boldsymbol{x}(n|n)$ because ***SEI***(n) and ***VEI***(n) matrices are fused in the same way. The DKF been recently described corresponds to Figure 4-1.

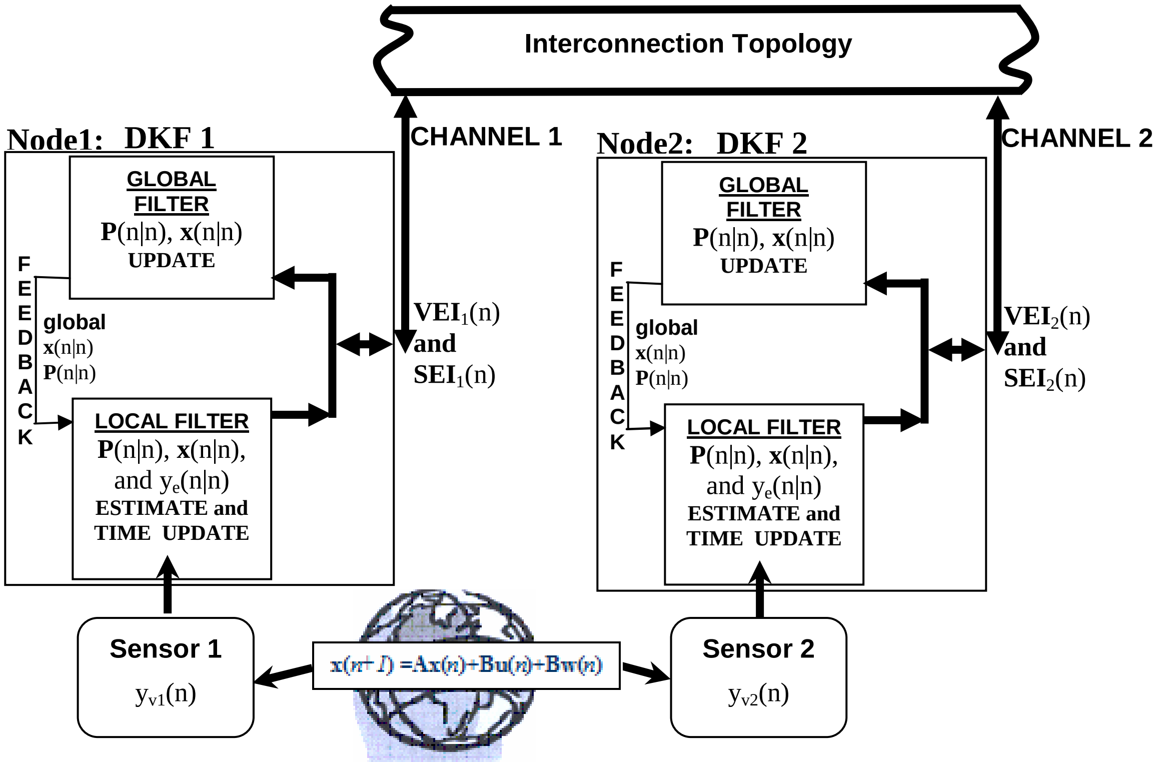
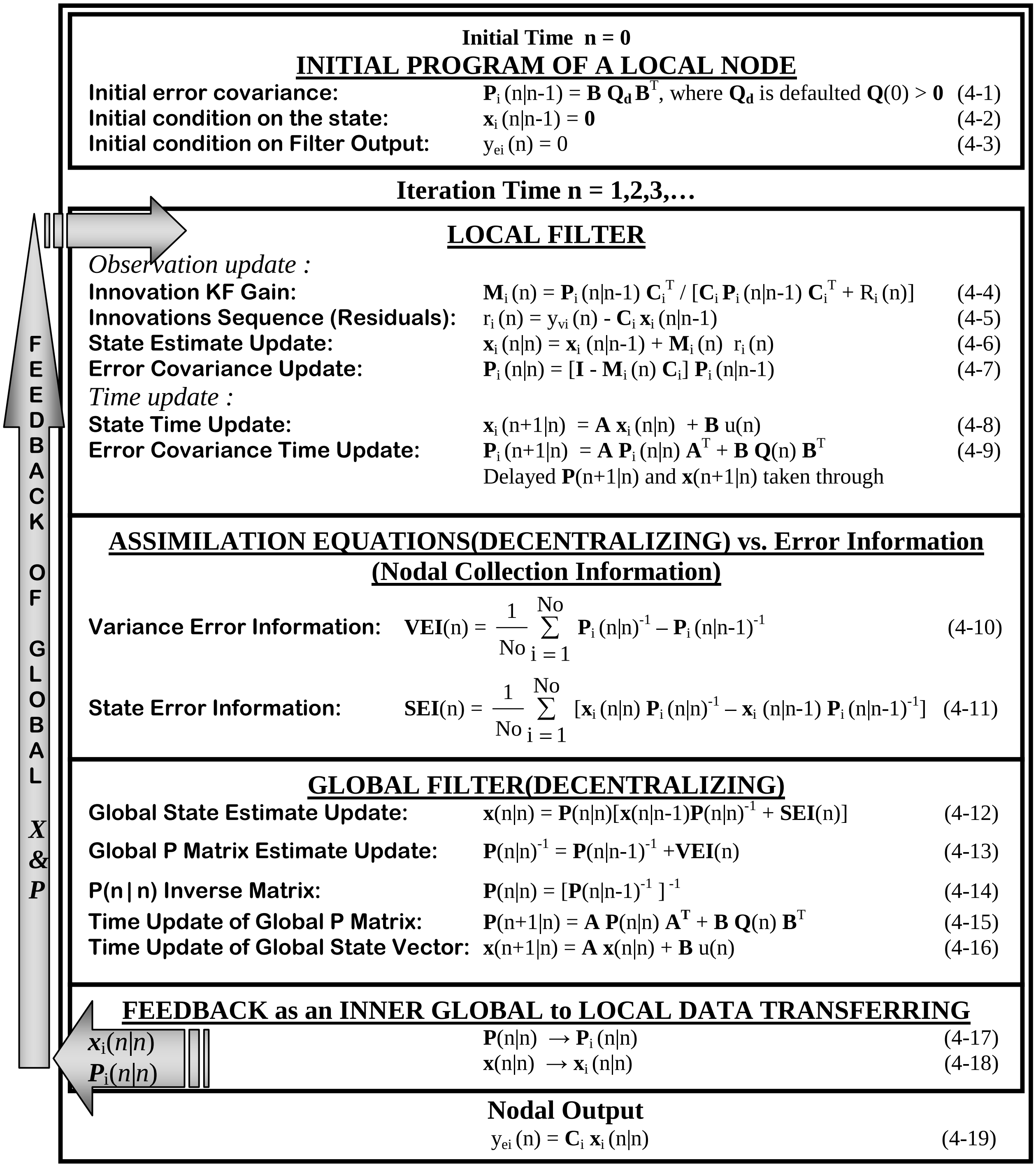


Figure 4-1 Decentralized Kalman filter topology.

Generally, the number of sensors is not restricted as well as Figure 4.1 described. We will describe the functionality of DKF system by the following flowchart in Table 4-1.



Initial Time n = 0

INITIAL PROGRAM OF A LOCAL NODE

Initial error covariance: $\mathbf{P}_i(n|n-1) = \mathbf{B}\,\mathbf{Q_d}\,\mathbf{B}^T$, where $\mathbf{Q_d}$ is defaulted $\mathbf{Q}(0) > \mathbf{0}$ (4-1)

Initial condition on the state: $\mathbf{x}_i(n|n-1) = \mathbf{0}$ (4-2)

Initial condition on Filter Output: $y_{ei}(n) = 0$ (4-3)

Iteration Time n = 1,2,3,…

LOCAL FILTER

Observation update :

Innovation KF Gain: $\mathbf{M}_i(n) = \mathbf{P}_i(n|n-1)\,\mathbf{C}_i^T / [\mathbf{C}_i\,\mathbf{P}_i(n|n-1)\,\mathbf{C}_i^T + R_i(n)]$ (4-4)

Innovations Sequence (Residuals): $r_i(n) = y_{vi}(n) - \mathbf{C}_i\,\mathbf{x}_i(n|n-1)$ (4-5)

State Estimate Update: $\mathbf{x}_i(n|n) = \mathbf{x}_i(n|n-1) + \mathbf{M}_i(n)\; r_i(n)$ (4-6)

Error Covariance Update: $\mathbf{P}_i(n|n) = [\mathbf{I} - \mathbf{M}_i(n)\,\mathbf{C}_i]\,\mathbf{P}_i(n|n-1)$ (4-7)

Time update :

State Time Update: $\mathbf{x}_i(n+1|n) = \mathbf{A}\,\mathbf{x}_i(n|n) + \mathbf{B}\,u(n)$ (4-8)

Error Covariance Time Update: $\mathbf{P}_i(n+1|n) = \mathbf{A}\,\mathbf{P}_i(n|n)\,\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}(n)\,\mathbf{B}^T$ (4-9)

Delayed **P**(n+1|n) and **x**(n+1|n) taken through

ASSIMILATION EQUATIONS(DECENTRALIZING) vs. Error Information (Nodal Collection Information)

Variance Error Information: $\mathbf{VEI}(n) = \frac{1}{No}\sum_{i=1}^{No} \mathbf{P}_i(n|n)^{-1} - \mathbf{P}_i(n|n-1)^{-1}$ (4-10)

State Error Information: $\mathbf{SEI}(n) = \frac{1}{No}\sum_{i=1}^{No} [\mathbf{x}_i(n|n)\,\mathbf{P}_i(n|n)^{-1} - \mathbf{x}_i(n|n-1)\,\mathbf{P}_i(n|n-1)^{-1}]$ (4-11)

GLOBAL FILTER(DECENTRALIZING)

Global State Estimate Update: $\mathbf{x}(n|n) = \mathbf{P}(n|n)[\mathbf{x}(n|n-1)\mathbf{P}(n|n)^{-1} + \mathbf{SEI}(n)]$ (4-12)

Global P Matrix Estimate Update: $\mathbf{P}(n|n)^{-1} = \mathbf{P}(n|n-1)^{-1} + \mathbf{VEI}(n)$ (4-13)

P(n | n) Inverse Matrix: $\mathbf{P}(n|n) = [\mathbf{P}(n|n-1)^{-1}]^{-1}$ (4-14)

Time Update of Global P Matrix: $\mathbf{P}(n+1|n) = \mathbf{A}\,\mathbf{P}(n|n)\,\mathbf{A^T} + \mathbf{B}\,\mathbf{Q}(n)\,\mathbf{B}^T$ (4-15)

Time Update of Global State Vector: $\mathbf{x}(n+1|n) = \mathbf{A}\,\mathbf{x}(n|n) + \mathbf{B}\,u(n)$ (4-16)

FEEDBACK as an INNER GLOBAL to LOCAL DATA TRANSFERRING

$\mathbf{P}(n|n) \rightarrow \mathbf{P}_i(n|n)$ (4-17)

$\mathbf{x}(n|n) \rightarrow \mathbf{x}_i(n|n)$ (4-18)

Nodal Output

$y_{ei}(n) = \mathbf{C}_i\,\mathbf{x}_i(n|n)$ (4-19)

Table 4-1 Timing diagram of DKF.

The local filter of the *i*-th node computes its own local estimate updates (4-6) and (4-7) using residuals (4-5) and an innovation KF gain (4-4), respectively. The time updates (4-8) and (4-9) are also performed by a local filter of each *i*-th DKF node. The local filter of the *i*-th DKF node works independently and indirectly from the other nodes and all their residuals differ. After the local filter was processed, the ***SEI***(*n*) of (4-11) and ***VEI***(*n*) of (4-10) are computed by assimilation equations. So, every nodal DKF computes its own part such as the *i*-th part of either (4-10) or (4-11) that is called fractional matrix. These fractional matrices are sent both to other nodes where are collected to get global ***SEI***(*n*) and ***VEI***(*n*). The fractional matrices are meant to be expressed such as $\boldsymbol{x}_{\mathrm{i}}(n|n)\boldsymbol{P}_{\mathrm{i}}(n|n)^{-1}$ - $\boldsymbol{x}_{\mathrm{i}}(n|n\text{-}1)\boldsymbol{P}_{\mathrm{i}}(n|n\ \text{-}1)^{-1}$ for ***SEI***(*n*), and $\boldsymbol{P}_{\mathrm{i}}(n|n)^{-1}$ - $\boldsymbol{P}_{\mathrm{i}}(n|n\ \text{-}1)^{-1}$ for ***VEI***(*n*), where the subscript *i* denotes the *i*-th node and filter. The global filter of the *i*-th node computes global estimates by received ***SEI***(*n*) of (4-12) and ***VEI***(*n*) of (4-13). There the time updates of global state and covariance matrix are computed in (4-16) and (4-15), respectively.

In each processor (node), a feedback process is running, where the global filter sends a global updated estimates $\boldsymbol{x}(n|n)$ and $\boldsymbol{P}(n|n)$ covariance matrix into its own local filter. This way, the $\boldsymbol{x}(n|n)$ and $\boldsymbol{P}(n|n)$ are interchanged with $\boldsymbol{x}_{\mathrm{i}}(n|n)$ and $\boldsymbol{P}_{\mathrm{i}}(n|n)$, respectively. It refers to (4-17) and (4-18). Finally, the state estimate updates are evaluated in the same manner in all nodes. This fact tends to DKF data fusion, where the states are computed optimally. The global output can be obtained in (4-19).

Each DKF estimator can be embedded in a sensor. Besides, in CKF structure all observations y_{vi} need to be broadcasted to the corresponding *i*-th nodes. The advantage of DKF against CKF is an embedded processor of DKF in sensor, hence no central fusion is needed, see [12]. In DKF node the observations are used directly. In sense of estimation, the advantage of DKF is also the less sensitive estimator to corrupted ***SEI***(*n*) and ***VEI***(*n*) when corrupted broadcasting happens. In other words, each sensor node has its own processing element, and its own communication facilities.

The DKF algorithm has a number of important features [11]:

- Global estimates by all nodes are guaranteed to be exactly the same as those obtained by CKF.
- A failure of any broadcasted signal tends to estimate degradation. But it will not result a whole system failure. This fact works in opposite to centralized fusion.
- A small additional computation required.
- Low communication overhead than similar structures.

Each node must communicate one $\boldsymbol{SEI}(n)$ vector and one matrix $\boldsymbol{VEI}(n)$ to each other node. Assuming there are N_o sensing nodes and each node estimates a full state vector of dimension l, then a total of $(l^2 + l)(N_o\text{-}1)$ numbers need to be communicated in each cycle. In CKF systems (central fusion) only N_o numbers are equal to the number of sensors need to be broadcasted, [9].

There are some fundamental limitations of environment-based descriptions in a single source of sensor information toward to multi-sensor systems [13]:

- Single sensors can only provide partial information on their operating environment
- The observations made by a sensor are always uncertain and occasionally spurious or incorrect; single sensor systems have no way to reduce this uncertainty.
- Different sensors provide different types of information according to different tasks; but no single sensor can cover all tasks.
- The failure of single sensor results in complete system failure; they are not robust in critical systems.

4.1 Computational improvement

Always covariance matrix $\boldsymbol{P}(n|n)$ is being singular and equal to $\boldsymbol{BQ}(n)\boldsymbol{B}$ at the beginning of simulation when n = 0. The problem is that $\boldsymbol{Q}(n)$ is set to diagonal matrix in the term of $\boldsymbol{BQ}(n)\boldsymbol{B}$ at n = 0. In such cases, this flowchart in Figure 4.1-1 gives enhancement, in addition to stability of performance in the fact that singularity of $\boldsymbol{BQ}(n)\boldsymbol{B}$ term is overcome. The matrix $\boldsymbol{O}$ is defined in Section 3.1. The (4-13) and (4-14) are treated by Figure 2.1-1 and (4-12) is treated by Figure 3.1-1.

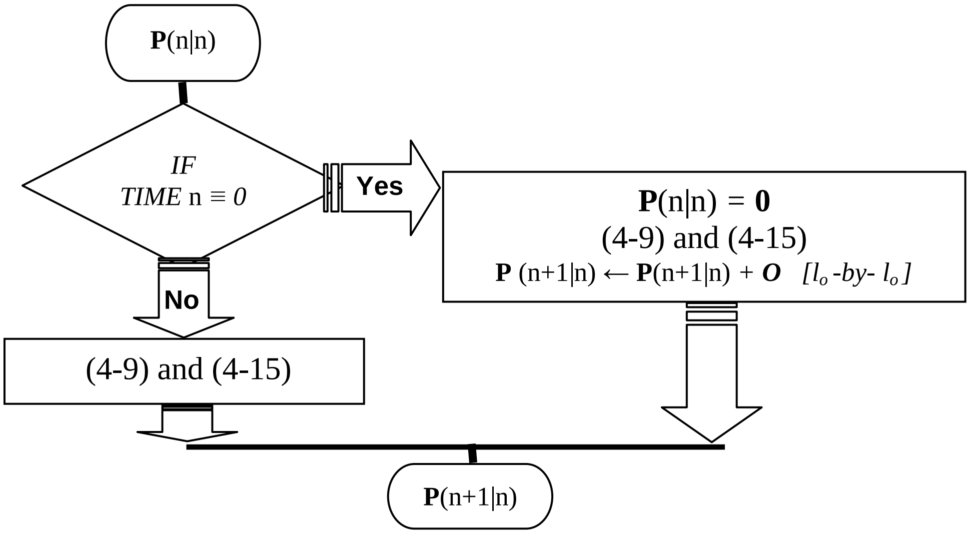


Figure 4.1-1 Treatment of the DKF error covariance time update calculation.

5 ADAPTIVE KALMAN FILTER TECHNIQUES

This unit gives a survey to methodology and models of an adaptive Kalman filter namely AKF with one sensor; adaptive centralized Kalman filter, ACKF; and adaptive decentralized Kalman filter, ADKF in multi-sensor fusion. Three appropriate timing diagrams for mentioned models would be presented. These models are coupled with further in unit 13. The reference [17] is utilised.

At first, we will focus on how to specify an adaptive process and subsequently flowcharts of the above mentioned types of KF models. A method of adaptive Kalman filtering, AKf, is called *adaptive* due to the fact that one KF parameter at least should be adapted, for instance sensor noise variance $R_a(n)$. The good way, of how to verify and adapt KF whether if its covariance is correct, is residuals monitoring.

$$r(n) = y_v(n) - \boldsymbol{C}\,\boldsymbol{x}(n \mid n-1) \tag{5-1}$$

The innovations sequence (residuals) (5-1) is a difference between the by a sensor measured observations $y_v(n)$ out of the plant (2-2) and the priori observations $\boldsymbol{Cx}(n|n\text{-}1)$ of estimator.

$$S(n) = R_a(n) + \boldsymbol{C}\,\boldsymbol{P}(n \mid n-1)\boldsymbol{C}^T \tag{5-2}$$

The covariance of the innovation sequence called associated noise $S(n)$ is defined by (5-2), where the

$$\boldsymbol{C}\,\boldsymbol{P}(n \mid n-1)\boldsymbol{C}^T \tag{5-3}$$

is priori error covariance of filtered output, see [17]. To be honest, the above formula is evaluated in second stage of Kalman filter and the associated noise $S(n)$ is measured as a squared innovation sequence below

$$S(n) = r^2(n), \tag{5-4}$$

where the mean of $r(n)$ sequence meets the condition below:

$$E[r(n)] = 0\,. \tag{5-5}$$

Combining the (5-2) and (5-4), the observation noise covariance $R_a(n)$ is adapted and extracted easily. Additionally, an algorithm to adapt the process noise covariance matrix may be established and used in some examples in Unit 13. Generally, the purposed adaptive method is useful for noise covariance matrix estimation, elsewhere possibly for time varying process. The innovation sequence plays an important role in the AKf and can be selected as an indicator of some actual estimation imperfections.

The AKF determines, within its calculation procedure, whether the covariance matrix of estimation error is considered as a theoretical idea of estimation accuracy or not. In practical applications, a gap can occur between the actual estimation errors and theoretically predicted ones. This can be explained by the fact that all applied system models are never exactly correct and can be affected by time-varying process in noises covariance. Moreover in all types of Kalman filters, the monitoring of the innovation sequence can be employed in fault detection systems. The method of AKf relies on the noise stream statistics but not on any unknown sensor failure. If a failure occurs, the Kalman filter could follow it as well. This monitoring tends to work with catastrophic sensor failure without any fault detection and isolation – FDI system. It will improve dramatically our understanding further in unit 12.

Adaptive Kalman filter. In this part, a timing diagram of AKF is presented in case of state estimation in single sensor fusion.

	<u>Initial Program</u> **Initial Time n = 0**	
Initial error covariance:	$\mathbf{P}(n \mid n-1) = \mathbf{B}\,\mathbf{Q_d}\,\mathbf{B}^T$, where defaulted $\mathbf{Q_d} > \mathbf{0}$	(5-6)
Initial condition on the state:	$\mathbf{x}(n \mid n-1) = \mathbf{0}$	(5-7)
Initial condition on Filter Output:	$y_e(n) = 0$	(5-8)
	Iteration Time n = 1,2,3,…	
Observation update :		
Innovations Sequence (Residuals):	$r(n) = y_v(n) - \mathbf{C}\,\mathbf{x}(n \mid n-1)$	(5-9)
Power of Associated Noise:	$S(n) = r^2(n)$	(5-10)
Adapting $R_a(n)$:	$R_a(n) = \left\| S(n) - \mathbf{C}\,\mathbf{P}(n \mid n-1)\mathbf{C}^T \right\|$	(5-11)
Innovation KF Gain:	$\mathbf{M}(n) = \mathbf{P}(n \mid n-1)\;\mathbf{C}^T \Big/ \left(\mathbf{C}\;\mathbf{P}(n \mid n-1)\mathbf{C}^T + R_a(n)\right)$	(5-12)
State Estimate Update:	$\mathbf{x}(n \mid n) = \mathbf{x}(n \mid n-1) + \mathbf{M}(n)\; r(n)$	(5-13)
Error Covariance Update:	$\mathbf{P}(n \mid n) = [\mathbf{I} - \mathbf{M}(n)\mathbf{C}]\mathbf{P}(n \mid n-1)$	(5-14)
Estimated Filter Output:	$y_e(n) = \mathbf{C}\,\mathbf{x}(n \mid n)$	(5-15)
Error Covariance:	$\text{error cov} = \mathbf{C}\,\mathbf{P}(n \mid n)\mathbf{C}^T$	(5-16)
Time update :		
State Time Update:	$\mathbf{x}(n+1 \mid n) = \mathbf{A}\,\mathbf{x}(n \mid n) + \mathbf{B}\,u(n)$	(5-17)
Error Covariance Time Update:	$\mathbf{P}(n+1 \mid n) = \mathbf{A}\,\mathbf{P}(n \mid n)\;\mathbf{A}^T + \mathbf{B}\,\mathbf{Q_a}(n)\;\mathbf{B}^T$	(5-18)

Table 5-1 Adaptive Kalman filter.

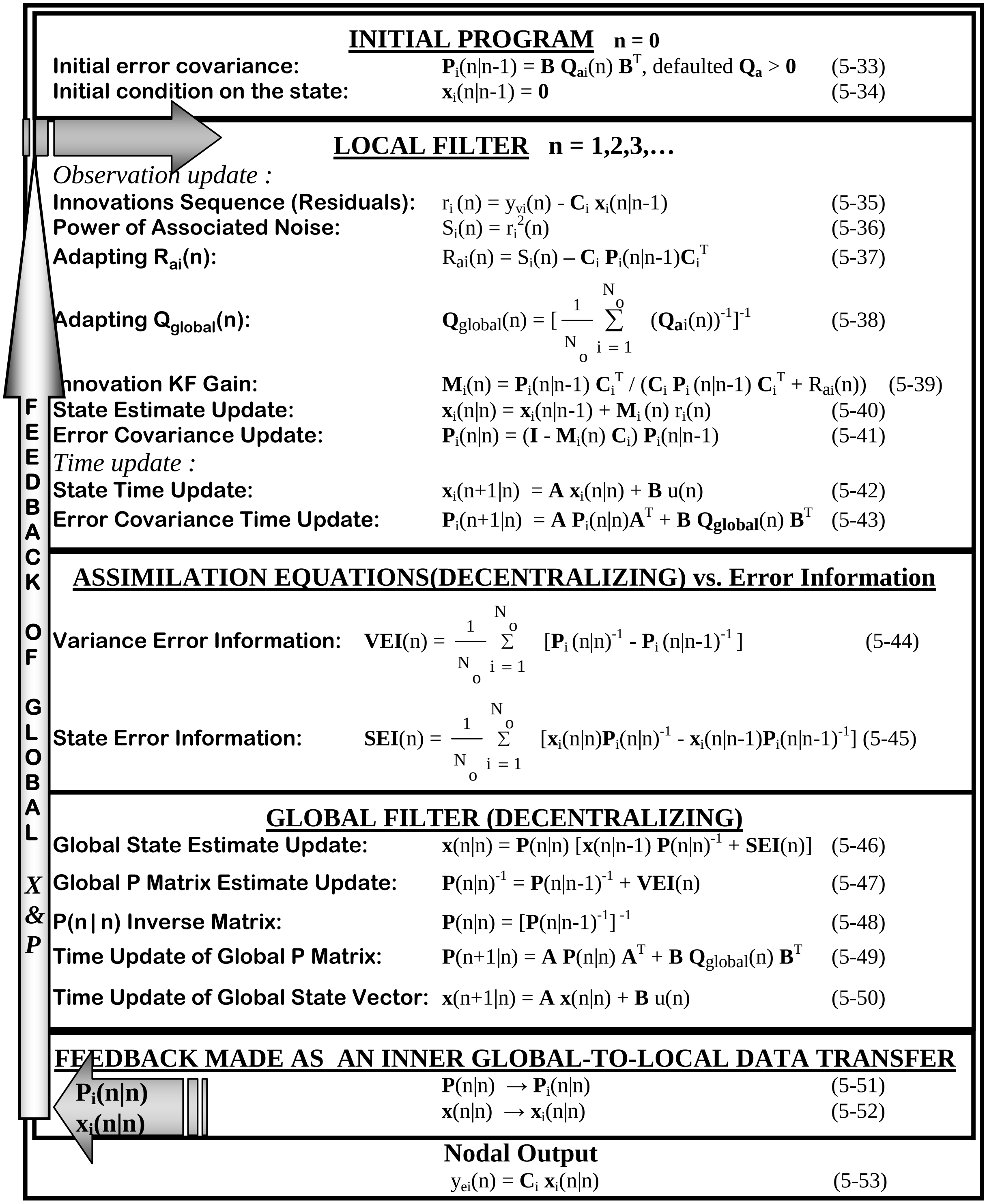
Adaptive Centralized Kalman Filter. Table 5-2 describes ACKF in multi-sensor fusion.

Initial Program **Initial Time n = 0**		
Initial error covariance:	$\mathbf{P}(n \mid n-1) = \mathbf{B}\,\mathbf{Q}_{\mathbf{d}}\,\mathbf{B}^{T}$, where defaulted $\mathbf{Q}_{\mathbf{d}} > 0$	(5-19)
Initial condition on the state:	$\mathbf{x}(n \mid n-1) = \mathbf{0}$	(5-20)
Initial condition on Filter Output:	$y_{ei}(n) = 0$	(5-21)
Iteration Time n = 1,2,3,…		
Innovations Sequence (Residuals):	$r_i(n) = y_{vi}(n) - \mathbf{C}_i\,\mathbf{x}(n \mid n-1)$	(5-22)
Power of Associated Noise:	$S_i(n) = r_i^{2}(n)$	(5-23)
Adapting $R_{ai}(n)$:	$R_{ai}(n) = \lvert S_i(n) - \mathbf{C}_i\,\mathbf{P}(n \mid n-1)\mathbf{C}_i^{T}\rvert$	(5-24)
Adapting $Q_{global}(n)$:	$\mathbf{Q}_{\mathbf{global}}(n) = \left[\frac{1}{N_o}\sum_{i=1}^{N_o}\mathbf{Q}_{\mathbf{ai}}^{-1}(n)\right]^{-1}$	(5-25)
Innovation KF Gain:	$\mathbf{M}_i(n) = \mathbf{P}(n \mid n-1)\mathbf{C}_i^{T} \Big/ \left[\mathbf{C}_i\,\mathbf{P}(n \mid n-1)\mathbf{C}_i^{T} + R_{ai}(n)\right]$	(5-26)
State Estimate Update:	$\mathbf{x}(n \mid n) = \mathbf{x}(n \mid n-1) + \frac{1}{N_o}\sum_{i=1}^{N_o}\mathbf{M}_i(n)\,r_i(n),\; N_o \geq 1$	(5-27)
Error Covariance Update:	$\mathbf{P}(n \mid n) = \left[\mathbf{I} - \frac{1}{N_o}\sum_{i=1}^{N_o}\mathbf{M}_i(n)\mathbf{C}_i\right]\mathbf{P}(n \mid n-1)$	(5-28)
Estimated Filter Output:	$y_{ei}(n) = \mathbf{C}_i\,\mathbf{x}(n \mid n)$	(5-29)
Error Covariance:	$\text{error cov}_i = \mathbf{C}_i\,\mathbf{P}(n \mid n)\mathbf{C}_i^{T}$	(5-30)
State Time Update:	$\mathbf{x}(n+1 \mid n) = \mathbf{A}\,\mathbf{x}(n \mid n) + \mathbf{B}\,u(n)$	(5-31)
Error Covariance Time Update:	$\mathbf{P}(n+1 \mid n) = \mathbf{A}\,\mathbf{P}(n \mid n)\,\mathbf{A}^{T} + \mathbf{B}\,\mathbf{Q}_{global}(n)\,\mathbf{B}^{T}$	(5-32)

Table 5-2 Adaptive centralized Kalman filter.

In this model used the same equations from Table 3-1 are used with (5-23) to (5-25). The global matrix $\boldsymbol{Q}_{global}(n)$ is inversely proportional to summarised matrices $\boldsymbol{Q}_{ai}^{-1}(n)$, see (5-25) and [3]. The $\boldsymbol{Q}_{ai}^{-1}(n)$ matrices are defined by adaptive process. The certain number of sensors is noted by N_o.

Adaptive Decentralized Kalman Filter. The initial program, (5-19) to (5-21), is used additionally.



INITIAL PROGRAM **n = 0**

Initial error covariance: $\mathbf{P}_i(n|n-1) = \mathbf{B}\,\mathbf{Q}_{ai}(n)\,\mathbf{B}^T$, defaulted $\mathbf{Q}_a > \mathbf{0}$ (5-33)

Initial condition on the state: $\mathbf{x}_i(n|n-1) = \mathbf{0}$ (5-34)

LOCAL FILTER **n = 1,2,3,…**

Observation update :

Innovations Sequence (Residuals): $r_i(n) = y_{vi}(n) - \mathbf{C}_i\,\mathbf{x}_i(n|n-1)$ (5-35)

Power of Associated Noise: $S_i(n) = r_i^2(n)$ (5-36)

Adapting $R_{ai}(n)$: $R_{ai}(n) = S_i(n) - \mathbf{C}_i\,\mathbf{P}_i(n|n-1)\mathbf{C}_i^T$ (5-37)

Adapting $Q_{global}(n)$: $\mathbf{Q}_{global}(n) = [\frac{1}{N_o}\sum_{i=1}^{N_o} (\mathbf{Q}_{ai}(n))^{-1}]^{-1}$ (5-38)

Innovation KF Gain: $\mathbf{M}_i(n) = \mathbf{P}_i(n|n-1)\,\mathbf{C}_i^T / (\mathbf{C}_i\,\mathbf{P}_i(n|n-1)\,\mathbf{C}_i^T + R_{ai}(n))$ (5-39)

State Estimate Update: $\mathbf{x}_i(n|n) = \mathbf{x}_i(n|n-1) + \mathbf{M}_i(n)\,r_i(n)$ (5-40)

Error Covariance Update: $\mathbf{P}_i(n|n) = (\mathbf{I} - \mathbf{M}_i(n)\,\mathbf{C}_i)\,\mathbf{P}_i(n|n-1)$ (5-41)

Time update :

State Time Update: $\mathbf{x}_i(n+1|n) = \mathbf{A}\,\mathbf{x}_i(n|n) + \mathbf{B}\,u(n)$ (5-42)

Error Covariance Time Update: $\mathbf{P}_i(n+1|n) = \mathbf{A}\,\mathbf{P}_i(n|n)\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}_{global}(n)\,\mathbf{B}^T$ (5-43)

ASSIMILATION EQUATIONS(DECENTRALIZING) vs. Error Information

Variance Error Information: $\mathbf{VEI}(n) = \frac{1}{N_o}\sum_{i=1}^{N_o} [\mathbf{P}_i(n|n)^{-1} - \mathbf{P}_i(n|n-1)^{-1}]$ (5-44)

State Error Information: $\mathbf{SEI}(n) = \frac{1}{N_o}\sum_{i=1}^{N_o} [\mathbf{x}_i(n|n)\mathbf{P}_i(n|n)^{-1} - \mathbf{x}_i(n|n-1)\mathbf{P}_i(n|n-1)^{-1}]$ (5-45)

GLOBAL FILTER (DECENTRALIZING)

Global State Estimate Update: $\mathbf{x}(n|n) = \mathbf{P}(n|n)\,[\mathbf{x}(n|n-1)\,\mathbf{P}(n|n)^{-1} + \mathbf{SEI}(n)]$ (5-46)

Global P Matrix Estimate Update: $\mathbf{P}(n|n)^{-1} = \mathbf{P}(n|n-1)^{-1} + \mathbf{VEI}(n)$ (5-47)

P(n|n) Inverse Matrix: $\mathbf{P}(n|n) = [\mathbf{P}(n|n-1)^{-1}]^{-1}$ (5-48)

Time Update of Global P Matrix: $\mathbf{P}(n+1|n) = \mathbf{A}\,\mathbf{P}(n|n)\,\mathbf{A}^T + \mathbf{B}\,\mathbf{Q}_{global}(n)\,\mathbf{B}^T$ (5-49)

Time Update of Global State Vector: $\mathbf{x}(n+1|n) = \mathbf{A}\,\mathbf{x}(n|n) + \mathbf{B}\,u(n)$ (5-50)

FEEDBACK MADE AS AN INNER GLOBAL-TO-LOCAL DATA TRANSFER

$\mathbf{P}(n|n) \rightarrow \mathbf{P}_i(n|n)$ (5-51)

$\mathbf{x}(n|n) \rightarrow \mathbf{x}_i(n|n)$ (5-52)

Nodal Output

$y_{ei}(n) = \mathbf{C}_i\,\mathbf{x}_i(n|n)$ (5-53)

Table 5-3 Adaptive decentralized Kalman filter.

New equations (5-35) - (5-37) are implemented into a local filter based on DKF. The matrix $\boldsymbol{Q}_{ai}(n)$ is transferred to every node of ADKF to obtain the $\boldsymbol{Q}_{global}(n)$. This part of fusion is performed by (5-38).

6 SIMULATION OF KALMAN FILTER

The purpose of this unit is CoKF verification and a performance presentation of this estimator. In addition to Section 2.1 intent for *Model 2 of KF*, a functionality of proposed model would be checked up before the use in real appliances. By the way, we want to find some model imperfections if any exist. So, the *Models 1, 2 and 3 of KF* would be verified. Additionally, a numerical approach to system model, etc., is defined in this unit and consequently used up to further units of this thesis to make it consistent. The first question given in Section 1.1 will be solved.

First of all, we are now about to put some conditions and signal definition on a system. A stochastic system model of third order filter can be described as follows:

$$\boldsymbol{x}(n+1)=\begin{bmatrix} 1.1269 & -0.4940 & 0.1129 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\boldsymbol{x}(n)+\begin{bmatrix} 0.3832 \\ 0.5919 \\ 0.5191 \end{bmatrix}[u(n)+w(n)]; \tag{6-1}$$

$$y_v(n)=[1 \quad 0 \quad 0]\boldsymbol{x}(n)+v(n), \tag{6-2}$$

where *w*(*n*) and *v*(*n*) is the process and observation white (zero mean) noise, respectively. This is the third order filter of linear finite-dimensional stochastic system model defined in discrete time domain. According to system (2-1) and (2-2), with regard to system dynamics, the ***A***; ***B*** and ***C*** matrices are certainly given in (6-1) and (6-2). Constant variances of process and sensor noise are namely given by:

$$\boldsymbol{Q}=\begin{cases} 3.669 \quad 10^{-2}\,\boldsymbol{I}, \text{ high power of a process noise w(n)} \\ 3.669 \quad 10^{-4}\,\boldsymbol{I}, \text{low power of a process noise w(n)} \end{cases}; \tag{6-3}$$

$$R = 8.69 \quad 10^{-4}, \tag{6-4}$$

where process noise *w*(*n*) has two options as a power. Starting this unit, the (6-3), (6-4) and corresponding wave form of noise streams are used in the all thesis regarding to no correlation of *w*(*n*) and *v*(*n*) noises. Those noises are assigned to realistic approach, where both are usually dominated in a low frequency band as shown in Figure 6-1.

The ***Q*** is diagonal process noise covariance matrix with (6-3) numbers. Simplifying the notation and having the ***Q*** matrix, diagonal numbers in the matrix are denoted by *Q*.

The *Q* to *R* ratio gives two options as 40 and 0.4 approximately that compliance the (6-3) and (6-4).

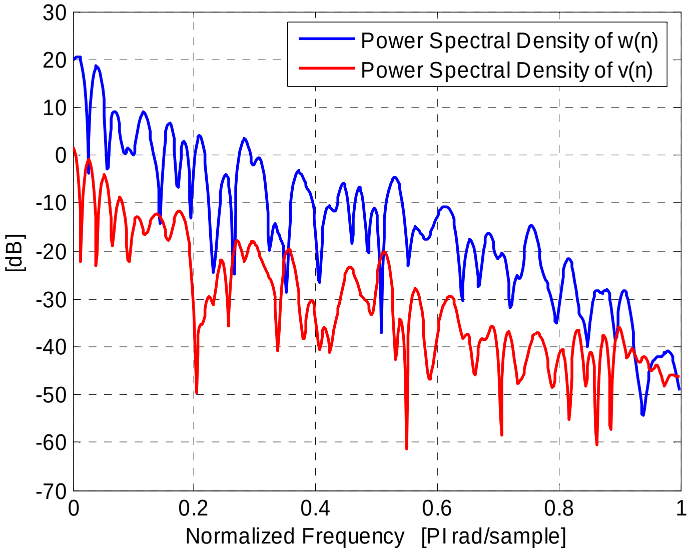


Figure 6-1 Power spectral density of w(n) and v(n).

To be honest, a frequency domain and waveform of *w*(*n*) and *v*(*n*) noises can be simulated, where the *w*(*n*) in not directly measured. The noises will have to fulfil the demands on limited frequency band, however the KF theory for optimal control suggests white noise as a wide approach.

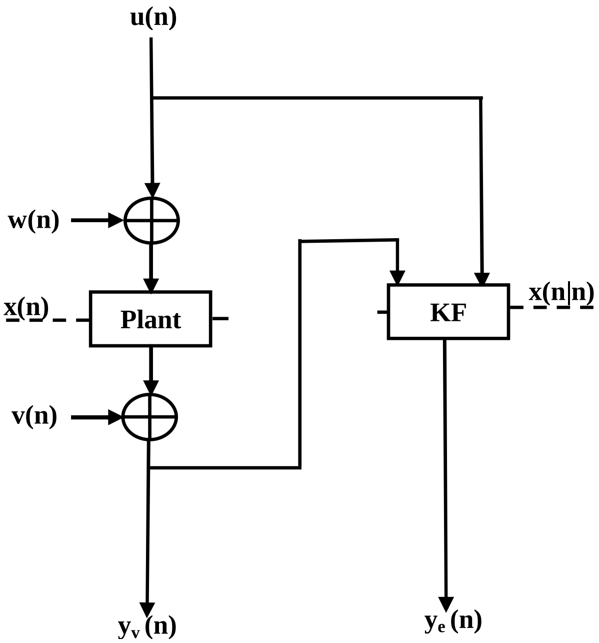


Figure 6-2 Plant state estimation proposed with Kalman filter.

After the system model was presented, the schema of estimation principle with plant and single-sensor fusion is shown in Figure 6-2. The state vector ***x***(*n*) is coupled with plant and evaluated using (6-1).

The state estimation noted by $\boldsymbol{x}(n|n)$ is only processed in the block of KF. The process noise such as an uncertainty is used in (6-1) and sensor noise model occurs in (6-2) of time-invariant system model. The sensor noise is also called the observation noise, $y_v(n)$ is plant output, and $y_e(n)$ is filtered KF output. Let us define the input control signal $u(n)$ sinus wave below:

$$u(n) = \sin\left(\frac{2\pi \quad n}{50}\right) . \tag{6-5}$$

Following formula is given for calculation a mean square error MSE. In order to evaluate the KF performance, the MSE factor will be used to measure state estimation accuracy.

$$MSE = E\left[\boldsymbol{a}^2(n)\right] - \left\{E\left[\boldsymbol{a}(n)\right]\right\}^2 \tag{6-6}$$

The substituted variable $\boldsymbol{a}(n)$ will be applied in an experimental study and results the formula (6-7), where a total number of summarised items can be of course either than 101 which will be stated.

$$\frac{1}{101}\sum_{n=0}^{100} \boldsymbol{x}^T(n)\boldsymbol{x}(n) \tag{6-7}$$

The Figure 6-1 shows the option of high power of Q (6-3), while R stands unchanged (6-4) in thesis.

The system model (6-1)-(6-2) and the principle carry out in Figure (6-2) are used up to Unit 11.

The noise streams and their variances (6-3), (6-4) and MSE formula (6-6) will be used along with next simulations up to Unit 14. In order to make multi-sensor fusion, many uncorrelated sensor noise streams should be used in a sensor model but variances of each of them remain still the same regarding to (6-4) up to the end of this thesis.

The input control signal (6-5) would be also used in further up to the Unit 12.

In this unit, our concern tends to find a mathematical identity and numerical similarity between CoKF models in some tests. We are focusing on three CoKF models called *Model* 1-3 *of KF* taken from the Unit 2. These models shown in Table 2-1 to 2-3 give their own filter observations called $y_{e1}(n)$, $y_{e2}(n)$, $y_{e3}(n)$, respectively only in tests of this unit. First two models give their own state estimations called $\boldsymbol{x}_1(n|n)$ and $\boldsymbol{x}_2(n|n)$, respectively. Last two models called *Model 2 of KF* and *Model 3 of KF* are mathematically derived from and based on equations of the first *Model 1 of KF*, hence all of them should be mathematically identical. To confirm this, they will be submitted in some tests.

State estimation accuracy will proceed in next steps as a function of process noise in examination on Kalman filtering.

Test for MSE of difference between $x_1(n|n)$ and $x_2(n|n)$. In this sub-test the MSE is shown in Table 6.1. The difference between state estimates done in first two models is measured optionally with low power of process noise and assigned to $\boldsymbol{a}(n)=[\boldsymbol{x}_1(n\,|\,n)-\boldsymbol{x}_2(n\,|\,n)]$.

Power of state x(n) of Plant	132.5
(MSE of [x1(n|n)]-x2(n|n)]) / (Power of x(n))	$8.5\ .\ 10^{-14}$

Table 6.1 Test of state estimation of Model 1 and 2 of KF.

Test for MSE of difference between $x(n) - x_1(n|n)$ and $x_2(n|n)$. The aim is to measure a signal difference between expecting state and estimated one. Hence the evaluation criteria is $\boldsymbol{a}(\mathrm{n})=[\boldsymbol{x}(\mathrm{n})-\boldsymbol{x}_1(n\,|\,n)]$. The MSE for first model is shown in table.

Power of state x(n) of Plant	132.5
(MSE of [x(n)]-x1(n|n)]) / (Power of x(n))	$1.284\ .\ 10^{-4}$

Table 6.2 Test of state estimation of Model 1 of KF.

The accuracy of state estimation of first model is high because MSE is equal to 1.284_x10^{-4} and MSE for second model is quite same with small difference about 10^{-10}. Some values of $\boldsymbol{x}(n)$, $\boldsymbol{x}_1(n|n)$ and $\boldsymbol{x}_2(n|n)$ are presented below to complete Table 6.2.

x(n) of Plant	$[1.122;\ 1.733;\ 1.520]\ .\ 10^{-3}$
x1(n|n) of Model 1 of CoKF	$[3.157;\ 0.075;\ 3.190]\ .\ 10^{-3}$
x2(n|n) of Model 2 of CoKF	$[3.159;\ 0.076;\ 3.190]\ .\ 10^{-3}$

Table 6.3 Estimated state vectors of Model 1 and 2 of KF and plant at time n = 2.

Either both models perform with numerical small error presented by Tables 6.1, 6.2 and high accuracy of state estimation. Therefore the time n = 2 was chosen for Table 6.3 to make values unlike and visible with three decimal points as much as possible. The states $\boldsymbol{x}_1(n|n)$ and $\boldsymbol{x}_2(n|n)$ tend to $\boldsymbol{x}(n)$ in the worst case at the beginning of simulation when time n = 2, otherwise $\boldsymbol{x}_1(n|n)$ and $\boldsymbol{x}_2(n|n)$ are same.

Test for MSE of difference between observations. Here $y_{e3}(n)$ and $y_v(n)$ should be correlated with up to unwanted sensor noise, where variance of a difference between them is equal to 2.49_x10^{-4} and getting numerically close to 3.669_x10^{-4} variance of sensor noise. The differences between first two models and the third model are about 10^{-9} and 10^{-7}, respectively.

Moreover, the system model defined by (6-1) and (6-2) is presented in [5], and there the *Q*, *R* are equal to one for *w*(*n*) and *v*(*n*) random noises, and $u(n) = \sin\left(\frac{n}{5}\right)$. We do not measure MSE of state estimates in the case of *Model 3 of KF*, because this model does not estimate $\boldsymbol{x}(n|n)$ but it does $\boldsymbol{x}(n|n\text{-}1)$ only. Hence plotted $y_e(n)$ outputs are shown in Appendix A.1 corresponding to identical figure of first model in [5].

6.1 Conclusions

The Table 6.1 presents very small relative MSEs of *Model 1-3 of KF*. It means that all equations of the Unit 2 and the improvement of Section 2.1 are useful.

Small difference, 8.5_x10^{-14} in Table 6.1, between *Model 1* and *2 of KF* occurs due to singularity avoided by the needed method presented in Section 2.1.

High state estimation accuracy is presented with small relative MSE, 1.284_x10^{-4} in Table 6.2, in case of *Model 1* and *2 of KF*. Also in Table 6.3, the values of state vectors are shown at time *n* = 2. There the estimated states tend to state vector of plant numerically with small error. Assuming the periods of time when $n > 2$, state estimation accuracy of a KF is so high that only high number of digits is required to see any difference between states of plant and estimated states.

The third model shows figure of Appendix A.1 to be identical to one in [5]. The third model does not produce state estimates but it does only the filtered output $y_v(n)$. There the filtered output of the third model $y_{e3}(n)$ and observation $y_v(n)$ are correlated up to unwanted sensor noise *v*(*n*).

In a case of high power of process noise we can obtain similar conclusions. Measured relative MSE of difference between $\boldsymbol{x}_1(n|n)$ and $\boldsymbol{x}_2(n|n)$ equals to 1.155_x10^{-16} respecting the method presented above. There the state estimation accuracy is high with small relative MSE (4.5_x10^{-4}) that corresponds to *Model 1* and *2 of KF* for Table 6.2. A variance between observations of all three models is lower than the previous case of low power of process noise.

As notice, to make relevant simulation the value of $\boldsymbol{Q}$ must be truly set to the variance of process noise. Finally, high $\boldsymbol{Q}$ gathers the innovation KF gain $\boldsymbol{M}(n)$ resulting an amplified residuals $\boldsymbol{r}(n)$. So the optimal feedback between a plant and estimator is established. In case of small power, the error in state estimation yields 82% (3.669/4.5) out of power of *w*(*n*), and for high power the error of estimation is about 0.35% (1.284_x10^{-4} / $3.669\ _x10^{-2}$).

In this way, three models of the Unit 2 have been presented. They are mathematically derived correctly and numerically performing state estimation with high accuracy with thanks to the improvement presented in Section 2.1. This conclusion refers to the first solution in Section 1.3.

7 SIMULATION OF CENTRALIZED KALMAN FILTER

The aim of this unit is dedicated to verify CKF models built beyond MATLAB/Simulink. Two Sections, 7.1 and 7.2, include some simulation results. This would satisfy the first question of Section 1.1 with some successful Kalman filter models.

First, we specify system model by (6-1) and (6-2).

Referring back to the equations (6-3) and (6-4), certain matrices of *w*(*n*) and *v*(*n*) are used from the Unit 6. In all simulations the only high power of *w*(*n*) noise is considered, view page 26.

Additionally the input signal *u*(*n*) is defined by (6-5) , where $\mathrm{n} = 0,\ 1,\ \ldots,\ 100$.

Then an evaluation of CKF models would be performed by MSE calculation with formula (6-6), where

$$\boldsymbol{a}(n) = \boldsymbol{x}(n) - \boldsymbol{x}(n\,|\,n)\ . \tag{7-1}$$

A relative MSE will be computed so that the MSE is divided by a state power of plant (6-7). For the purpose of all simulations done up to Unit 11 the observation matrix can be given as follows:

$$\boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}. \tag{7-2}$$

The Section 7.1 refers to a simple simulation of *Model 1* to *3 of CKF*. Prior to any usage, this simulation will verify three models. After, when our CKF models are allowed for any use, they might be implemented into real devices. The CKF models are tested to find all failures and model imperfections even if any exist. The simulation is based on CoKF and CKF comparison, where two sensors are assumed and their sensor noise streams are totally identical that does not occur in reality. To be detailed, observation $y_v(n)$ is doubled for the reason of two nodes in CKF model. Hence this situation can be only simulated. This way we will prove a CKF fusion and a performance of presented equations in the Unit 3. Due to the fusion, a number of sensors should not change **x**(*n*|*n*) if all sensors signals and particular noises are equal. So, we expect the state vector **x**(*n*|*n*) produced by CKF with two sensors to be on an equality with **x**(*n*|*n*) of single sensor CoKF estimator, and also their equal filtered $y_e(n)$ outputs. To make the fusion correct, we the ∑ - sum is displaced in material [15] by E - mean.

The Section 7.2 refers to simulation with *Model 1* to *3 of CKF* containing two nodes. Although there is no comparison between CoKF and CKF because two independent sensors are used in CKF model, and their corresponding sensor noise streams are uncorrelated and zero-mean.

7.1 Testing the CKF fusion and performance

The CoKF model and CKF models are built according to equations of Table 2-1 and Tables 3-1 to 3-3, respectively. Same conditions for CoKF and CKF are used as in the Unit 6.

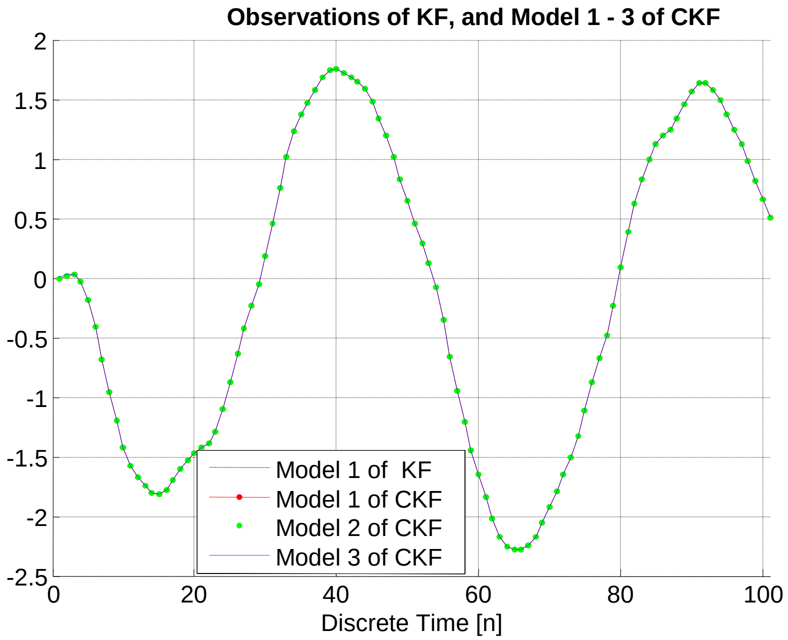


Figure 7.1-1 The CKF and KF observations of first output.

In Figure 7.1-1, the filtered outputs of CKFs and CoKF are overlapped without any difference. In other words, the outputs of CKF and CoKF are the same quality and mathematically equivalent in this test.

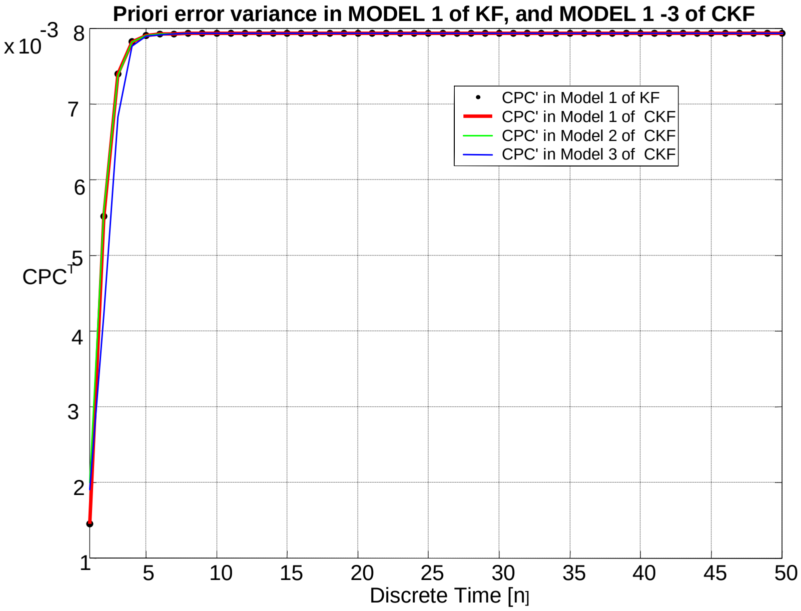


Figure 7.1-2 Priori error variance CP(n|n-1)C^T of filtered outputs in CKF and CoKF.

The Figure 7.1-2 shows the priori error variance of filtered outputs computed by $\boldsymbol{CP}(n|n\text{-}1)\boldsymbol{C}^T$. Finally, all curves tend asymptotically to constant. On the basis of comparability with all models, the covariance time update performs quite the same. One difference occurs at the discrete time $n = 0$ when inverse covariance matrix is singular, but this defect is overcome by improvements presented in the Sections 2.1 and 3.1. Then the singularity problem has no longer effect on state estimation performance. In the case of CoKF estimator, the MSE is shown in Table 7.1-1 in second row and the relative MSE in last one. The power of state $\boldsymbol{x}(n)$ is equal to 1.076.

SNR of State of Plant	14.83 dB
MSE of CoKF	$4.940 \,.\, 10^{-4}$
(MSE of CoKF) / (**x**(n) power of Plant)	$4.593 \,.\, 10^{-4}$

Table 7.1-1 MSE for Model 1 of KF.

The same MSE values are obtained for *Model 1* and *2 of CKF*. The first item presents the power of state from a noiseless plant, which is divided over an extracted noise from the noisy plant state $\boldsymbol{x}(n)$. This ratio is measured with decibel scale. A model of noiseless plant is created in our simulation but it does not occur in a real system.

First model of CoKF and CKF is described by fundamental equations and fusion works correctly. Those equations, as program controls, are listed in Table 2-1 and 3-1. The applied CKF fusion collects nodal information of both nodes to be summarized into global information and divided by number of sensors that equals to two. In order to present results of third CKF model, see Table 7.1-2 in below.

SNR of State of Plant	14.83 dB
MSE of CKF	$4.936 \,.\, 10^{-4}$
(MSE of CKF) / (**x**(n) power of Plant)	$4.589 \,.\, 10^{-4}$

Table 7.1-2 MSE for Model 3 of CKF.

The *Model 3 of CKF* gives MSE about $4.936_{x}10^{-4}$ and Table 7.1-1 of KF gives $4.94_{x}10^{-4}$ of MSE. Those values are getting numerically very close with the difference about $4_{x}10^{-7}$. This is caused due to a singularity problem in inverse covariance matrix calculation pending second iteration in this simulation. The *Model 3 of CKF* includes three computations of inverse covariance matrix and the *Model 2 of CKF* includes only two. Therefore, the *Model 3 of CKF* is computationally more demanded than the previous *Model 1 of KF* called CoKF.

7.2 Testing the CKF based on two sensors

In this section, we consider *Model 1* to *3 of CKF* but CoKF is not needed here. The aim is to compare the *Model 2* and *3 of CKF* toward the *Model 1 of CKF*. The *Model 3 of CKF* is fully centralized structure of KF and useful to solve the first question of Section 1.1 in Unit 8. Therefore it will be submitted next test to confirm good CKF performance.

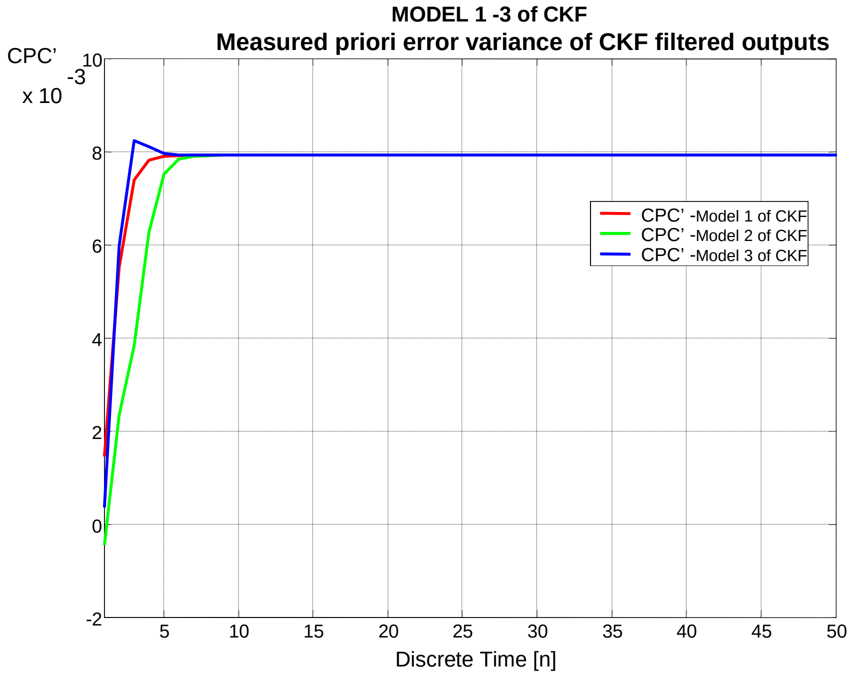


Figure 7.2-2 Priori error variance CP(n|n-1)C^T of all three CKF models.

The Figure 7.2-2 shows the priori error variance $\boldsymbol{CP}(n|n\text{-}1)\boldsymbol{C}^T$ of filtered outputs. All curves tend to a constant value. Additionally, the covariance time update performs comparably in all models. The state error covariance matrix $\boldsymbol{P}(n|n\text{-}1)$ becomes singular during first two discrete times. The effect of improvements can be seen in the beginning of Figure 7.2-2. After that the matrix is computed correctly and $\boldsymbol{CP}(n|n\text{-}1)\boldsymbol{C}^T$ becomes stable. The MSE of CKF models is computed in the same way with previous section. First two models of CKF give the same MSE value as shown in Tables 7.2-1.

SNR of State of Plant	14.83 dB
MSE of CKF	3.675 . 10^{-4}
(MSE of CKF) / (**x**(n) power of Plant)	3.416 . 10^{-4}

Table 7.2-1 MSE for Model 1 of CKF.

To notice the rule, a low MSE corresponds to high state estimation accuracy. The MSE of third model in Table 7.2-2 yields very likely value to the MSE of first model in Table 7.2-1.

SNR of State of Plant	14.83 dB
MSE of CKF	$3.679 . 10^{-4}$
(MSE of CKF) / (**x**(n) power of Plant)	$3.420 . 10^{-4}$

Table 7.2-2 MSE for Model 3 of CKF.

Small difference between them is caused by the complexity problem calculation of inverse covariance matrix especially computed at time $n = 1$ and 2. The best way of how to show high accuracy of state estimation directly is the demonstration of $\boldsymbol{x}(n|n)$ in CKF models at time $n = 2$, see Table 7.2-4 below.

x(n|n) of Model 1 of CKF	$[1.835; -1.685; -2.233] . 10^{-2}$
x(n|n) of Model 2 of CKF	$[1.835; -1.685; -2.233] . 10^{-2}$
x(n|n) of Model 3 of CKF	$[1.886; -9.775; -2.130] . 10^{-2}$

Table 7.2-4 State vectors of CKF models at discrete time n = 2.

7.3 Conclusions

After two tests in Sections 7.1 and 7.2, the CKF models are recognised to be usable in reality.
In Section 7.1, the CKF models are designed correctly. There the fusion of all CKF models performs the same state estimate with the regard to small error about 4_x10^{-7}. This model imperfection does not affect state estimation at all. In Tables 3-1 to 3-3 the fusion is performed by the following information sum $\frac{1}{N_o}\sum$. This sum is implemented in some equations of CKF model. The Section 7.1 verifies the reason of allowed total number of sensors N_o to be used such as waiting factor in the fraction denominator.
In Section 7.2, three CKF models perform numerically quite the same state estimation measured with MSE about 3.7_x10^{-4}. The measured values of state vector elements get numerically very close to state vector of plant $\boldsymbol{x}(n)$. The state estimation accuracy of CKF model was tested.
This conclusion joints with the importance of first solution in Section 1.3. The point of all Unit 7 is to obtain well performing CKF model with its correct fusion referred to the Section 7.1.

8 CENTRALIZED AND DECENTRALIZED KALMAN FILTER COMPARISON

In this unit we will dedicate the effort to explore identity between decentralized, DKF, and centralized, CKF, Kalman filter which can be measured by mean square error MSE in state estimates in an experimental simulation study. The MSE values of CKF and DKF models will be compared although both models are derived mathematically, see [11]. Both models are created according to Table 3-3 and 4-1 as a flow of control, and articles [9], [11-13]. The models consist of two nodes, shown in Appendix C.1.

The vast constraints are remarked on page 28, see Unit 6. The matrix ***C*** is defined in (7-2). The time *n* is running from 0 to 100 in MATLAB simulation. High power of $w(n)$ is been recently applied, (6-3).

Additionally, a relative MSE of a difference between DKF and CKF state estimates $\boldsymbol{x}(n|n)$ is shown at last row of the following table. The value in first row is related to difference between $\boldsymbol{x}(n)$ and $\boldsymbol{x}(n|n)$ of CKF state estimates. The second item explains a relative MSE of DKF computed in such way as the previous MSE in the first row.

(MSE of Model 3 of CKF) / ($\mathbf{x}^2$(n) of Plant)	3.420 . 10^{-4}
(MSE of DKF) / ($\mathbf{x}^2$(n) of Plant)	3.416 . 10^{-4}
(DKF **x**(n|n) - Model 3 of CKF **x**(n|n))^2 / ($\mathbf{x}^2$(n) of Plant)	3.429 . 10^{-7}

Table 8-1 MSE of DKF; MSE of CKF; MSE of estimated state **x**(n|n) (DKF-CKF) comparison.

Both values are very similar to satisfy the numerical identity among CKF and DKF. The error comparison is not crucial in performance comparison of both estimators that is still to come.

To make relevant comparison on estimators is to compute a difference between state vectors $\boldsymbol{x}(n|n)$ of DKF and CKF, see the last value. This relative MSE is about 10^3 times lower than the first two values. Actually, this value is expected to be small, because the DKF and CKF provide mathematically equal states. As an example, the Table 8-2 shows some sampled values of state vectors at $n = 2$.

x(n|n) of Model 3 of CKF	[-1.457; -0.719; -0.061]
x(n|n) of DKF	[-1.457; -0.719; -0.061]
x(n) of Plant	[-1.471; -0.729; -0.057]

Table 8-2 State vectors of Model 3 of CKF, DKF and plant at n = 2

One can see no numerical difference in state elements of both estimators due to low precision. The last item shows the $\boldsymbol{x}(n)$ of plant that is on numerical comparability with $\boldsymbol{x}(n|n)$ of CKF and DKF.

In our simulation, two sensors and two nodes of DKF estimator called DKF1 and DKF2 are shown. As shown in the last item of Table 8.1, the relative MSE is measured in the case of DKF1 and DKF2 and shown below in Table 8-3.

$\sum[(\text{DKF1 } \mathbf{x}(n\|n) - \text{CKF } \mathbf{x}(n\|n))^2] / (\mathbf{x}^2(n) \text{ of Plant})$	$3.429\ .10^{-7}$
$\sum[(\text{DKF2 } \mathbf{x}(n\|n) - \text{CKF } \mathbf{x}(n\|n))^2] / (\mathbf{x}^2(n) \text{ of Plant})$	$3.429\ .10^{-7}$

Table 8-3 DKF verification toward to CKF result based on stated measurement.

So, the DKF and CKF operate well despite very small difference measured in state estimates.

8.1 Conclusions

The first question of Section 1.1 is applied to this conclusion being answered below :

“Demonstrate state estimation identity of decentralized Kalman filter, DKF, and centralized Kalman filter, CKF, in an experimental simulation study assuming uniform filter conditions. What are differences?”

We have measured the relative MSE about 10^{-2} in state estimates of CKF and DKF. Measured MSE of $\boldsymbol{x}(n|n)$ difference between DKF1, DKF2 and CKF is about 10^3 smaller than the value mentioned firstly. These facts can be deeply clarified via in Table 8-1 and 8-3.

In the practical point, we can say that both estimators perform on the same state estimation despite the unimportant difference in our MATLAB simulations. This fact is caused by a computational problem of covariance matrix singularity that can be overcome using the improvements in Sections 3.1 and 4.1. Said the improvements are needed however causing a little inconvenience as a numerical problem. Without the improvements either the DKF or CKF can not start up, that usually gives a singularity problem. This is certain constraint. Toward to the power of $u(n)$ and $x(n\,|\,n)$ from the Unit 6, the DKF is designed so that this error 3.429_x10^{-7} is negligible.

Here we have presented the answer of the first question given by Section 1.1. In the next tests we should build upon the MATLAB based model of DKF. In our simulations all signals are broadcasted without any unknown latency or transmission failure in multi-sensor fusion.

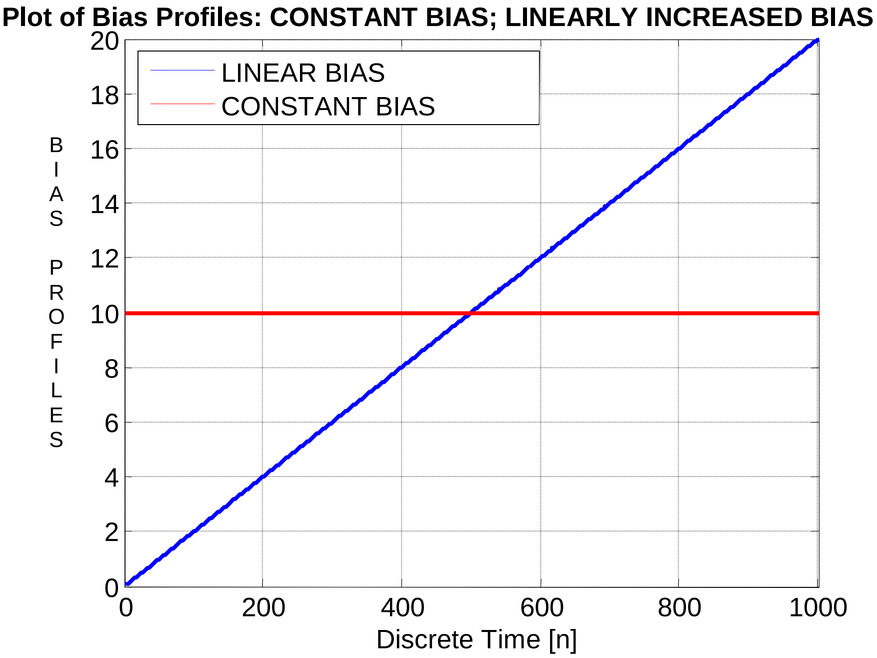
9 ONE BIASED SENSOR IN DKF MODEL

The aim of this unit is to make some experiments on DKF estimator that contains a sensor affected by a bias. Then an error seen on such affected state estimation would be measured in MATLAB. Thus a relation of a total number of sensors and MSE will be explored. The second question asked for in Section 1.1 will be answered. The Appendix C.3 and C.4 shows MATLAB models.

To make consistent work with previous units, the system model must be taken over the Unit 6. One possible selection is chosen when sensor noises, as $v_1(n)$; $v_2(n)$; etc., are uncorrelated, however their observation covariance $\boldsymbol{R}(n)$ are equal, see (6-4). As a simulation, the stream of $w(n)$ differs from the observation noises due to another pseudo random code. The MSE is defined by (6-6), $\boldsymbol{a}(n)$ by (7-1).

This unit is composed of four sections including some experiments with DKF performed with high and low power level of process noise, see (6.3). The first sensor noise is additively coupled with noise $v_1(n)$ and an existed bias. The Figure 9-1 illustrates time behaviours of bias.
First two sections (9.1 and 9.2) deal with DKF model that contains two and five sensors, respectively. One sensor is affected by constant bias over a time period of simulation. Following next two sections solve a problem of linearly increased bias considering two and five sensors.

The level of constant bias equals to 10, and the linear bias increases from zero up to 20.2 during simulation time, where $n = 0,\ 1,\ \ldots,\ 1000$.



9.1 Constant bias and two sensors in DKF model

A bias handling ability is identified by measuring the MSE of state estimation optionally with two power levels of process noise mentioned in tables below.

Constant Bias = 10	
MSE of DKF State Vector	1.014

Table 9.1-1 Two nodes decentralized Kalman filter result, high power of w(n).

Constant Bias = 10	
MSE of DKF State Vector	$4.312 . 10^{-2}$

Table 9.1-2 Two nodes decentralized Kalman filter result, low power of w(n)

The observations and filtered signals are shown in Figure 9.1-1. The setting of value in $\boldsymbol{Q}(n)$ must correspond truly to covariance of $w(n)$. If not so, the state estimation may be not optimal. Moreover as we can see in the figure, the bias influence works against the optimal state estimation. Making the $\boldsymbol{Q}(n)$ lower than the appropriate covariance of $w(n)$, the state estimation gives low accuracy while the bias impact on green curve is best. This way both terms can not be solved, therefore other useful technique is needed, see Unit 12.

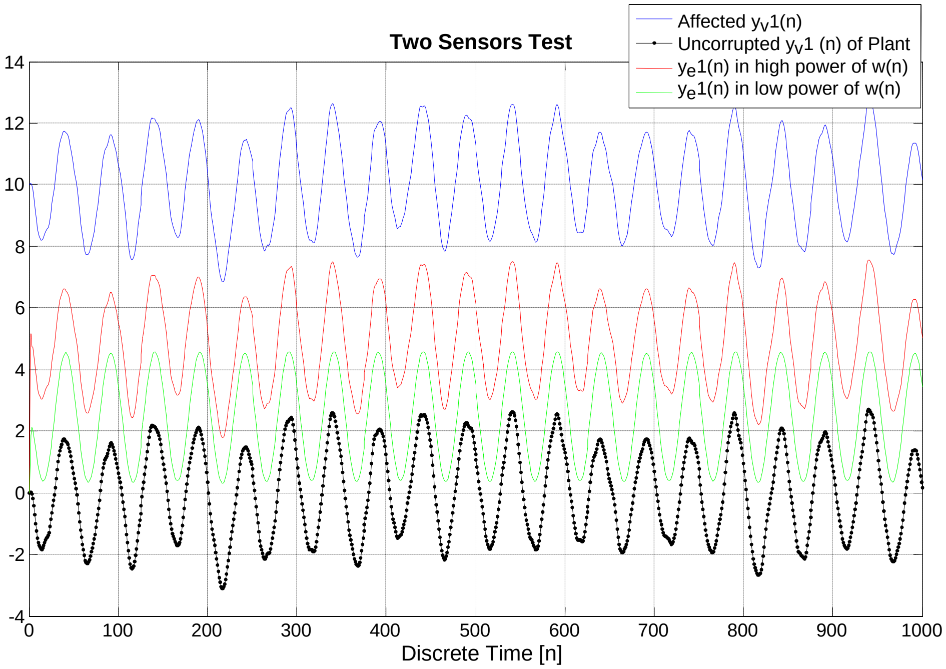


Figure 9.1-1 Observations in DKF model of constant bias.

9.2 Constant bias and five sensors in DKF model

This experiment is related to the one presented in Section 9.1 but five sensors are taken. Here the ***C*** matrix has the size of [5x3] with all rows been the same as in (7-2).

Constant Bias = 10	
MSE of DKF State Vector	$1.622 \,.\, 10^{-1}$

Table 9.2-1 Five nodes decentralized Kalman filter result, high power of w(n).

Constant Bias = 10	
MSE of DKF State Vector	$7.033 \,.\, 10^{-3}$

Table 9.2-2 Five nodes decentralized Kalman filter result, low power of w(n).

Here the goal is to find a relation of MSE and the total number of sensors in DKF model affected by bias. We would like to compare the MSE_{2DKF} of DKF model consist of two sensors, Table 9.1-1, and five sensors MSE_{5DKF} in the Table 9.2-1. Hence the MSE_{2DKF} model is higher than MSE_{5DKF} and their ratio $\sqrt{\text{MSE}_{\text{2DKF}} / \text{MSE}_{\text{5DKF}}} = \sqrt{6.252} = 2.5$. The ratio of total number of sensors is 2.5, which gets numerically near the computed value.

In MATLAB simulation with low power of $w(n)$, the ratio is equal to 2.476 been near the ratio of total number of sensors. The point is that the fusion in DKF model works as an averaging sum without any intelligence as FDI, where the FDI method may be able to eliminate a sensor failure.

9.3 Linearly increased bias and two sensors in DKF model

Having one sensor affected by linearly increased bias, two test results will be presented here.

Linearly Increased Bias	
MSE of DKF State Vector	6.11

Table 9.3-1 Two sensors decentralized Kalman filter result, high power of w(n).

Linearly Increased Bias	
MSE of DKF State Vector	2.171

Table 9.3-2 Two sensors decentralized Kalman filter result, low power of w(n).

The Figure 9.3-1 shows the time behaviour of observations of plant and filtered output of DKF.

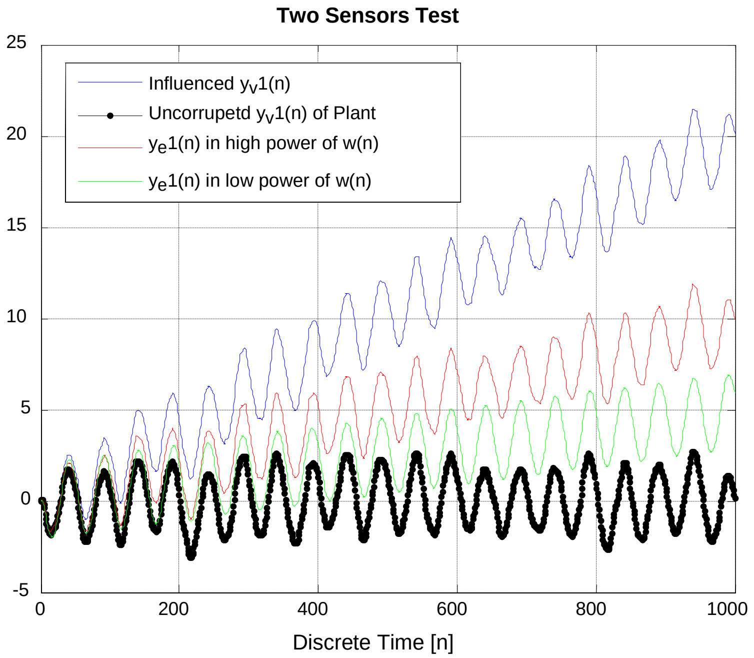


Figure 9.3-1 Observations in DKF model of linearly increased bias.

9.4 Linearly increased bias and five sensors in DKF model

In this section, same conditions are assumed and five sensors are keeping the same bias profile from previous section.

Linear Increased Bias	
MSE of DKF State Vector	$9.762 \,.\, 10^{-1}$

Table 9.4-1 Five sensors dentralized Kalman filter result, high power of w(n).

Linear Increased Bias	
MSE of DKF State Vector	$3.467 \,.\, 10^{-1}$

Table 9.4-2 Five sensors dentralized Kalman filter result, low power of w(n).

We are now about to compare the MSE_{2DKF} = 6.11 of DKF model composed of two nodes in Table 9.3-1 toward the MSE_{5DKF} = 9.762_x10^{-1} from the Table 9.4-1. Their ratio

$\sqrt{\mathrm{MSE}_{\mathrm{2DKF}} / \mathrm{MSE}_{\mathrm{5DKF}}}$ equals to $\sqrt{6.259} = 2.5$. This ratio is numerically equal to theoretical one given by five to two ratio of sensors number. High power of process noise has been applied too. In the same way of previous calculation, MSE ratio is about 2.502 been close to the theoretical value.

9.5 Conclusions

First the notice should be given so that one difference occurs since the third order filter as the system has been defined with two and five sensors optionally. The goal is to compare the $\mathrm{MSE}_{\mathrm{2DKF}}$ and $\mathrm{MSE}_{\mathrm{5DKF}}$ so that the $y_{v1}(n)$ to $y_{v5}(n)$ are same quality with $\boldsymbol{C}(1,n)\boldsymbol{x}(n)$, where additional sensor noise streams $v_1(n)$ to $v_5(n)$ differ one from the other.

To improve our understanding, refer to Appendix C.3 and C.4.

The second question of Section 1.1 is applied to this conclusion being answered below :

"Measure a mean square error, MSE, of state estimation in simulated DKF model that contains a sensor been affected by a bias with uniform filter conditions. Simulations have to be performed with total numbers of sensors such as two and five. How does the MSE depend on the total number of sensors? Tests need to be performed, namely with:

- constant bias,
- linearly increased bias."

In this experimental study we explored the problem of one biased sensor occurred in DKF model that contains two and five sensors as a total. Two types of bias were used in our simulations.

The relation of mean square error, MSE, and the total number of sensors is found valid for both types of bias. This relation can be explained using the example with two and five sensors models, where $\sqrt{\mathrm{MSE}_{\mathrm{2DKF}} / \mathrm{MSE}_{\mathrm{5DKF}}}$ is equal to number of sensors ratio given by $\frac{5}{2}$. The DKF model works as an averaging sum in fusion without any intelligence such as FDI, where the FDI method might be able to eliminate a sensor failure.

The power of plant state was 1.25, power of constant bias was 100 and power of linearly increased bias was 134. In other words, the power of bias was higher than the power of state. High power of bias may cause catastrophic state estimation in Kalman-Filter estimation.

The 100-times big change of $\boldsymbol{Q}(n)$ makes:

- 23-times changed MSE for the test with constant bias (compare values in Table 9.1-1 and 9.1-2, or Table 9.2-1 and 9.2-2);
- 2.8-times for the test with linear bias (Table 9.3-1 and 9.3-2, or Table 9.4-1 and 9.4-2);

Those MSE values differ due to the bias profiles, but the manner of bias time behaviour and effect on state estimation or filtered output remains the same.

10 ONE BROKEN SENSOR IN DKF MODEL

The purpose of this Unit is to obtain a relation of MSE of state estimation measured in DKF model and a total number of sensors. Only one sensor can be broken within total number of sensors. In our model simulations a broken sensor provides zero value instead of its eventual correct filtered output $y_v(n)$, where $\mathrm{n} = 0,\ 1,\ \ldots,\ 1000$. This zero provided value is accepted by DKF estimator.

The model of DKF comprises of two and five sensors used in Sections 10.1 and 10.2, respectively. In each section, two experiments are optionally performed with high and low power level of process noise. The first section shows some results of a problem mentioned above, where two sensors are associated with one system. The Section 10.2 presents some results with five sensors. Our MATLAB models belong to Appendix C.3 and C.4. The DKF is developed according to Table 4.1.

The system dynamics is defined in Unit 6 where the C matrix takes the form of (7-2) or large size as [5x2]. The MSE values and $\boldsymbol{a}(n)$ are computed according to formulas (6-6) and (7-1), respectively.

10.1 Testing the DKF with two sensors

This section presents two tables assuming different power levels of process noise noticed in below the Tables 10.1-1 and 10.1-2. This experiment yields two sensors in DKF model.

Broken Sensor	
MSE of DKF State Vector	$3.597 \ . \ 10^{-1}$

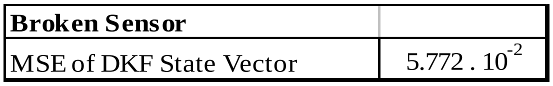
Table 10.1-1 DKF result in two sensors test of broken sensor, high power of w(n).

Broken Sensor	
MSE of DKF State Vector	$1.380 \ . \ 10^{-1}$

Table 10.1-2 DKF result in two sensors test of broken sensor, low power of w(n).

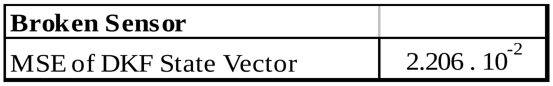
10.2 Testing the DKF with five sensors

In the same way with previous section, the MSE is measured on five sensors system. We are about to compare the MSE in Table 10.1-1 called MSE_{2DKF} and MSE_{5DKF} in Table 10.2-1.



Broken Sensor	
MSE of DKF State Vector	$5.772 \,.\, 10^{-2}$

Table 10.2-1 DKF result in five sensors test of broken sensor, high power of w(n).



Broken Sensor	
MSE of DKF State Vector	$2.206 \,.\, 10^{-2}$

Table 10.2-2 DKF result in five sensors test of broken sensor, low power of w(n).

There MSE_{2DKF} is higher than MSE_{5DKF} of five sensors model and their ratio $\sqrt{\text{MSE}_{\text{2DKF}} / \text{MSE}_{\text{5DKF}}}$ yields $\sqrt{6.232} = 2.496$. It is computed by the assumption of high power of $w(n)$ where its covariance is time-invariant. This ratio is numerically close to theoretical value given by $\frac{5}{2}$ as the five to two ratio (a ratio of total number of sensors). This rule is valid also in a case of low power of process noise with its covariance matrix $\boldsymbol{Q}$, where the $\sqrt{\text{MSE}_{\text{2DKF}} / \text{MSE}_{\text{5DKF}}}$ ratio is $\sqrt{6.256} = 2.501$.

10.3 Conclusions

Below we apply the third question of Section 1.1 to this conclusion to be answered:
“Measure the MSE of state estimation in simulated DKF model that contains a broken sensor with uniform filter conditions. Simulations have to be performed with two and five total numbers of sensors. How does the MSE depend on the total number of sensors?”

The tested DKF model is composed of two and five sensors, where only first of them was broken.

The ratio $\sqrt{\mathrm{MSE}_{2\mathrm{DKF}} / \mathrm{MSE}_{5\mathrm{DKF}}}$ equals to 2.496 and 2.501 assuming high and low power of process noise, respectively. Both values are numerically near to the expected value 2.5. This value could not be reached exactly because a power of sensor noise is not negligible and makes a small inaccuracy in calculation, however all observations of sensors contain same power of sensor noise in our simulations. From the measured MSE, noises are smoothed and a mean of them should be zero and any little mean does not affect MSE calculation at all. A noise plays an important role in calculation mentioned above because it is an unwanted random signal.

Comparing the values of MSE in experiments of corrupted sensor, Units 9 and 10, and correct sensor, see Unit 6, here the value rapidly increased due to a failure of sensor.

The third question asked in Section 1.1 for finding the relation between MSE and the total number of sensors is answered in this conclusion.

11 BY A DRIFT AFFECTED SENSOR IN DKF MODEL

This unit focuses on testing of estimators with sensors affected by a drift. All experiments present decentralized Kalman filter, DKF, as an estimator in multi-sensor fusion. The main point of corrupted system is to view an accuracy of state estimation as the opposite aspect of its measured mean square error MSE. This way we like to find a relation between MSE and a total number of sensors wherein only one of them would be affected by exponentially decreasing drift.

The DKF technique is implemented into all nodes of the decentralized structure simultaneously performing estimation, see Unit 4. In a general way, at least one sensor in any node might be affected by a drift. In all experiments of this unit, only first sensor is affected of which behaviour becomes additionally a subject of our study. This sensor is simulated as a observations of plant multiplied by drift varying from one down to zero. This type of sensor gives a fault in state estimation. The remaining last sensors are drift-free sensors which provide correct observations to be accepted in the structure of DKF.

Due to the drift, an optimal state of system may not be obtained. High state estimation accuracy corresponds to a low MSE of state estimation performed by DKF. Here the tests are beyond a simulation of DKF model that contains one, two and five sensors as the total number of sensors.

The drift problem in KF model with only one sensor is presented in Section 11.1. There we want to see drift based approach to a basic problem in KF processing. The *Model 1 of KF* is used.

Next Sections 11.2 and 11.3 include DKF experiments with two and five sensors, respectively.

Two options of process noise power are used. Some simulated models shown in Appendix C.2 to C.4 are related to Section 11.1 to 11.3. All proposed MATLAB simulations are done in time periods of $n = 0,\ 1,\ ...,\ 1000$; the MSE formula is given by (6-6), where differential signal $\boldsymbol{a}(n)$ is given by (7-1). As the assumption, variances of all sensor noises are equal but their noises are uncorrelated. Process noise remains the same in all simulations with the kind of drift and input signal. The system dynamics can be described as mentioned in the previous unit.

11.1 Kalman Filtering and drift in observations

First of all, it is needed to obtain some MSE measurements from the model of the drift. With the given system model the (6-2) should be redefined to

$$y_v(n) = d(n)\boldsymbol{Cx}(n) + v(n)\ . \tag{11.1}$$

This is a model of drifted sensor. Inside of the sensor, the drift might be not technically repaired directly. The exponential drift $d(n)$ can be defined as follows:

$$d(n) = e^{-\frac{n-\delta}{\Delta}};\ \delta = 0,\ \ \Delta = 400\,. \tag{11-2}$$

The drift slopes down starting the simulation when $n = 0$ and its exponent is determined by $\Delta = 400$. The (11-2) formula is used starting with this unit up to the end of the thesis.

We start this experiment with low power of process noise in the system. In the case of KF with one sensor (for CoKF), the effect on observations is shown in Figure 11.1-1. The filtered output $y_e(n)$ never reaches wrong value of zero while impacted observation $y_v(n)$ does up to noise $v(n)$. The drift gives majority in error of state estimation but KF is not designed to cope with drift kind of errors. Thus drift problem can not be solved by adjusting of $\boldsymbol{Q}$ matrix, hence this is related to Unit 12.

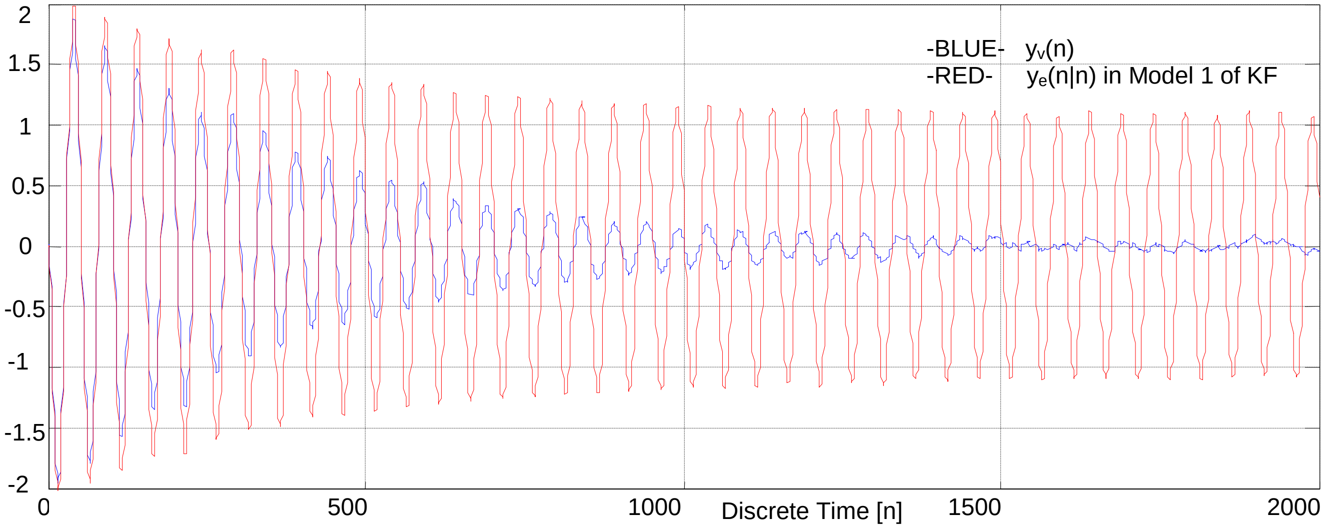
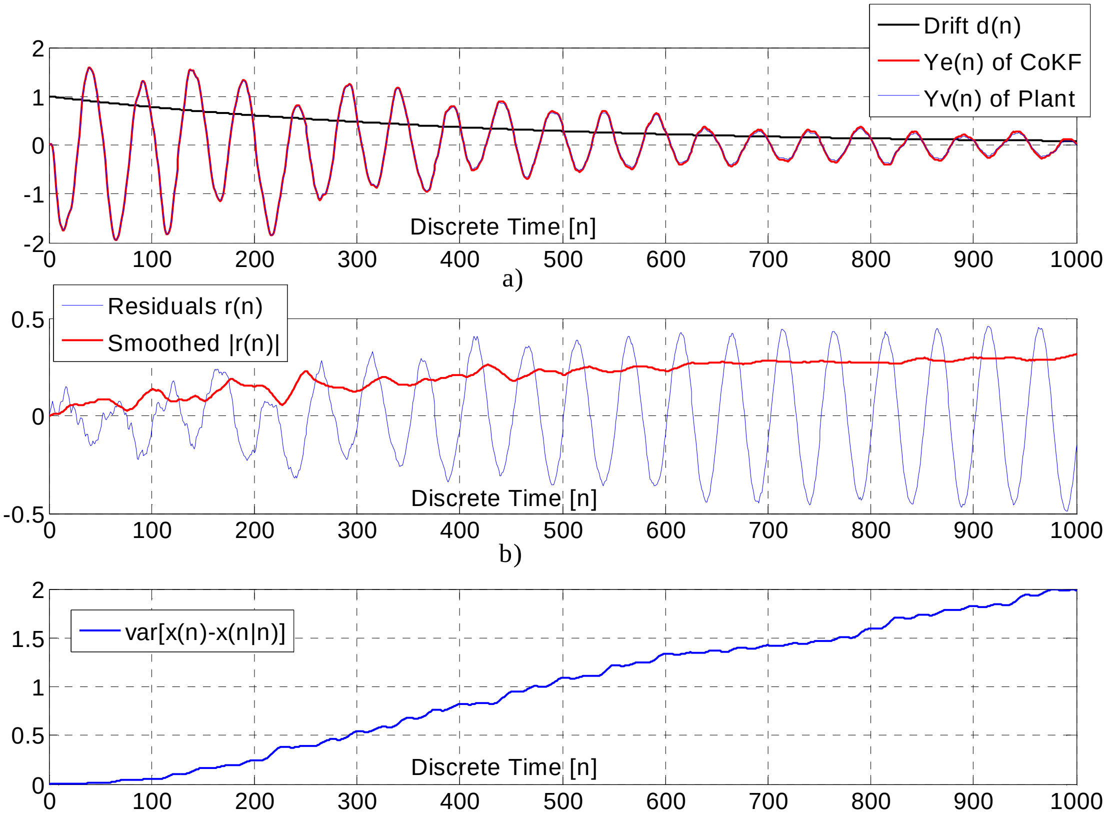


Figure 11.1-1 Filtered output of CoKF and observations of plant, low of process noise.

Making the result viewable, the simulation is long. The Figure 11.1-2 shows some measured signals optionally with high power of process noise. There the filtered output $y_e(n)$ of KF follows affected plant observation $y_v(n)$ and curves of both signals are shown in Figure 11.1-2a.



a)

b)

c)

Figure 11-1.2 The examples of drift in system with one sensor. In **a)** $y_e(n)$ of Kalman filter and $y_v(n)$ of plant; **b)** influenced residuals r(n) in blue and smoothed | r(n)|; **c)** variance measured between expected **x**(n) state of plant and **x**(n|n) estimated state of Kalman filter.

The goal is also to get an experience in residual $r(n)$ behaviour presented by curve of Figure 11.1-2b. As an important fact, smoothed absolute value $|r(n)|$ is correlated with the exponential drift shown in black of Figure 11.1-2a. Normally high residual values tend to an error in state estimation. Then, Figure 11.1-2c depicts the error variance between state of plant $\boldsymbol{x}(n)$ and estimator $\boldsymbol{x}(n|n)$. This variance is being increased during the appearing drift and refers to MSE in Table 11.1-1 and Table 11.1-2. Those tables present MSE of the same model assuming low power of process noise.

Drift Impact	
MSE of (CoKF State -Plant State)	6.618 . 10^{-1}

Table 11.1-1 MSE of difference between plants and estimators state, high level of process noise.

Drift Impact	
MSE of (CoKF State - Plant State)	2.572 . 10^{-1}

Table 11.1-2 MSE of difference between plants and estimators state, low level of process noise.

11.2 Drift and two sensors in DKF model

In this section, DKF model contains two sensors and first of them is only affected by the drift. Now we are allowed to compare some results of this and last section considering MSE values.

Drift Impact	
MSE of (DKF State - Plant State)	1.659 . 10^{-1}

Table 11.2-1 MSE of difference between plants and estimators state, high level of process noise.

Drift Impact	
MSE of (DKF State - Plant State)	6.423 . 10^{-2}

Table 11.2-2 MSE of difference between plants and estimators state, low level of process noise.

The MSE in Table 11.1-1 called MSE_{1KF} equals to 6.618_x10^{-1} and the MSE in Table 11.2-1 called MSE_{2DKF} equals to 1.659_x10^{-1} and their ratio $\sqrt{\text{MSE}_{1KF} / \text{MSE}_{2DKF}}$ is $\sqrt{3.989} = 1.997$. In similar way, measured ratio of values in Table 11.1-2 and Table 11.2-2 gives $\sqrt{4.004} = 2.001$.

This section can be concluded in fact that these results are near the expected value equal to $\frac{2}{1}$ as a ratio of total number of sensors. This small dissimilarity is caused by a little mean of noises $w(n)$ and $v(n)$.

11.3 Drift and five sensors in DKF model

In the same way with previous section, the MSE is measured and shown in following tables, but the model employs five sensors.

Drift Impact	
MSE of (DKF State - Plant State)	$2.686 . 10^{-2}$

Table 11.3-1 MSE of difference between plants and estimators state, high level of process noise.

Drift Impact	
MSE of (DKF State - Plant State)	$1.031 .10^{-2}$

Table 11.3-2 MSE of difference between plants and estimators state, low level of process noise.

We will compare MSE values of Sections 11.1 and 11.3. The MSE_{1KF} equals to 6.618_x10^{-1} and the MSE of DKF model with five sensors called MSE_{5DKF} equals to 2.686_x10^{-2}. Their ratio $\sqrt{MSE_{1KF} / MSE_{5DKF}}$ is $\sqrt{24.639} = 4.964$. In the case of low level of process noise, the ratio gives $\sqrt{24.947} = 4.995$. Our results tend to expected ratio given by five because one and five sensors have been taken into account with comparable quality in DKF model.

11.4 Conclusions

Below we apply the fourth question of section 1.1 to this conclusion to be answered:
“Measure the MSE of state estimation in DKF model that contains a sensor affected by a drift with uniform filter conditions. Simulations have to be performed by using one, two, and five total numbers of sensors. How does the MSE depend on the total number of sensors?”

As an approximation, the $\sqrt{MSE}$ is linearly proportional to the total number of sensors. For example in our case of two sensors model, the estimator gives four times lower MSE of state estimation than DKF with one sensor. Five sensors DKF provides 25-times lower MSE than the single sensor DKF.
The fourth question of finding the relation between MSE and total number of sensors has been answered in this conclusion.

12 FAULT DETECTION AND ISOLATION METHOD IN DECENTRALIZED KALMAN FILTERING

In Unit 11, we saw sensor drift problem that is expected to be solved by an additional system able to detect a drift as a sensors fault. The drift gives a fault in state estimation. Hence this unit deals with developing and simulating a traditional concept of fault detection and isolation, FDI, algorithm used in MATLAB simulations, where a nodal structure of decentralized Kalman filter is assumed and consist of three sensors/nodes. The FDI as an additional algorithm improves on DKF based structure to avoid sensors' failures. The DKF model with FDI system belongs to Appendix C.5.

Here the FDI algorithm would be used to detect a fault of an affecting sensor by an exponential drift and consequently isolate it to minimise an error caused by the fault in state estimation. The algorithm of FDI should be implemented in a self-acting technical system. The advantage of the algorithm is that every node of the decentralized structure will be able to detect and isolate fault in sensor its self. As the sensor information is propagated to all neighbouring nodes by a link interconnection, the FDI algorithm is simultaneously performing fault detection in each node to improve robustness. It allows us to implement the algorithm in a real time processing. The FDI technique is developed, [6], with an inspiration of the fault-tolerant design, and correlation based detection used to reach middle accuracy performance. Various techniques of fault detection for fault-tolerant design with diverse applications are reported in [6]. We will present the reliability and some imperfections of FDI.

Starting this unit, a new model of linear finite-dimensional stochastic system dynamics is used up to the end of this thesis as follows:

$$\boldsymbol{x}(n+1)=\begin{bmatrix} 2.633 & 1 & 0 \\ -2.33 & 0 & 1 \\ 6.907\ .\ 10^{-1} & 0 & 0 \end{bmatrix}\boldsymbol{x}(n)+\begin{bmatrix} 6.02\ .\ 10^{-2} \\ -1.132\ .\ 10^{-1} \\ 5.91\ .\ 10^{-2} \end{bmatrix}[u(n)+w(n)],\quad \boldsymbol{x}(0)=\boldsymbol{0}, \qquad (12\text{-}1)$$

$$\begin{bmatrix} y_{v1}(n) \\ y_{v2}(n) \\ y_{v3}(n) \end{bmatrix}=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\boldsymbol{x}(n)+\begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{bmatrix}. \qquad (12\text{-}2)$$

In simulations the $n = 0, \ 1, \ \ldots, \ 4000$. All noise description is taken over the Unit 6, excepting the control input. A time response of the observations behaves as a third order of low pass filter, and all observations $y_{v1}(n)$, $y_{v2}(n)$, $y_{v3}(n)$ give incomparable quality. Therefore the system model is technically suitable for its property in next units too. The third order filter is suitable in navigations, where three components such as a position, velocity and acceleration are estimated. Hence three components of $\boldsymbol{x}(n)$ refer to three sensors.

Thus the uncorrupted system dynamics has been introduced but no fault of a sensor has been presented yet. The equation (12-2) describes an ideal observations model without any fault.

After applying the drift problem, the fault in states will be solved in multi-state and multi-sensor DKF.

12.1 Testing the FDI implemented in DKF structure with one failed sensor

This section introduces the first example and the idea of FDI algorithm to be coupled with DKF model. Throughout this and next section, the (12-1) is considered but the observation equation (12-2) of stochastic model is omitted due to sensor fault and therefore a new realistic approach to the fault coupled with first sensor is found as follows:

$$\begin{bmatrix} y_{v1}(n) \\ y_{v2}(n) \\ y_{v3}(n) \end{bmatrix} = \begin{bmatrix} d_1(n) & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}(n) + \begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{bmatrix}, \tag{12-3}$$

where $d_1(n)$ is defined by (11-2) with $\delta = 1500$ and $\Delta = 200$. The delta δ describes the time when the exponential drift starts. Next parameter Δ describes the slope on how quickly the drift turns down inside of the first sensor.

We turn to observation processing of the first sensor because only this sensor is affected by the exponential drift. The Figure 12.1-1 shows the observation $y_{v1}(n)$ and time behaviour of drift in black.

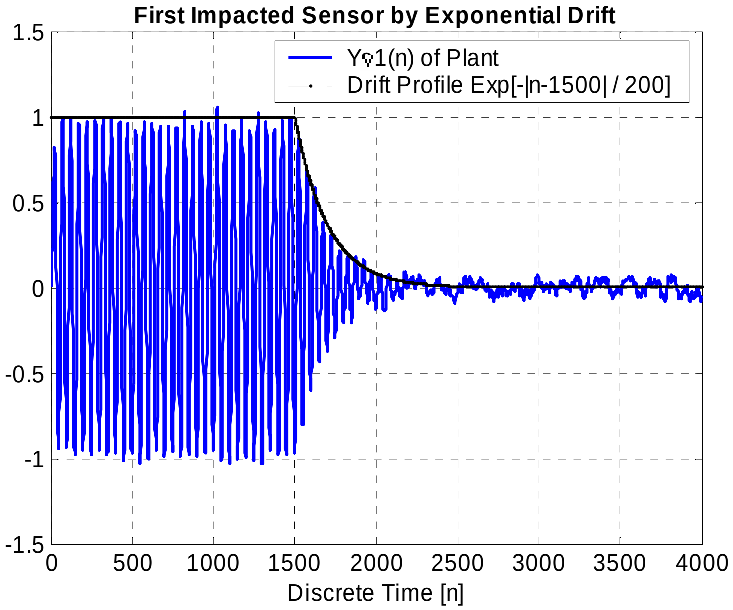


Figure 12.1-1 First corrupted sensor by exponential drift.

This drift effect provides a fault signal in residual depict in Figure 12.1-2. The red curve represents smoothed absolute value $|r(n)|$ of residual (innovations sequence) produced in first node.

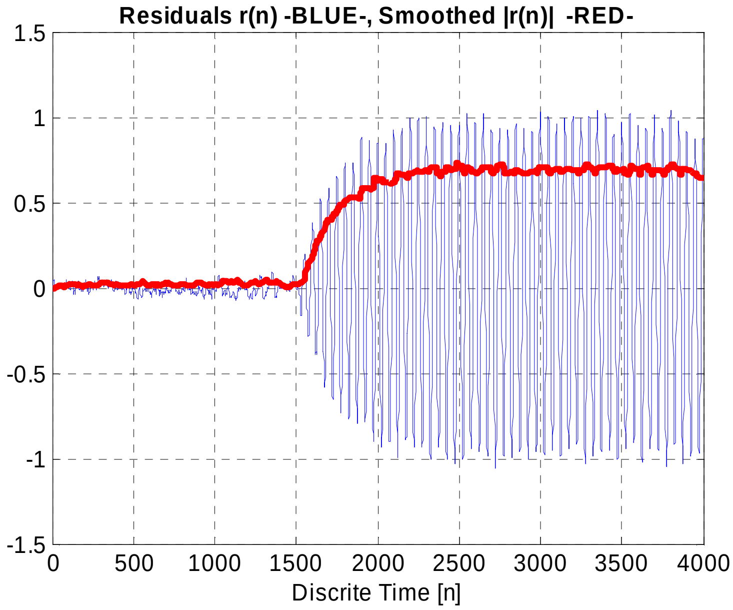


Figure 12.1-2 Residuals in blue, and smoothed $|r_1(n)|$ in red.

The estimators of second and third node have correct or a little noisy residuals $r_2(n)$ and $r_3(n)$. Signals of $r_2(n)$ and $r_3(n)$ are expected to be not correlated with $y_{v2}(n)$ and $y_{v3}(n)$ respectively, when at least another sensor is corrupted. The FDI algorithm attempts to reach zero moveable failure signal into the residuals of neighbouring nodes (i = 1 and 2). All three residuals are expected to be processed by DKF and FDI in a way to have a nominal, i.e. no-failure mode. In the absence of a failure, residuals should be close to the observation noises. Three different residuals is referred to three nodes that can be found as follows:

$$r_i(n) = y_{vi}(n) - \boldsymbol{Cx}(n \mid n), \tag{12-4}$$

and coupled with the ***i***-th sensor in ***i***-th node, where i = 1, 2, 3. Each node includes its own processor with DKF technique to perform state estimation in real time processing. The $\boldsymbol{x}(n|n)$ results on the same in every node of decentralized structure either with or without any sensor failure. The FD and consequential FI algorithms are applied in every processor to avoid or at least minimise a fault in state estimation.

As it has been outlined, the FDI deals with two algorithms such as fault detection and fault isolation. Both will be explored and simulated with comprehensive results in two parts below.

Fault detection. This part presents a general idea of the fault detection algorithm executed in every processor of corresponded the *i*-th node. It indicates that a fault is recognised, see the flowchart below as the timing diagram.

Fault Detection system inside of DKF$_i$ node

Preprocessing – Median Filter vs. Smoother

$$Y_i(n) = \mathrm{MEDIAN}\left\{\left|r_i(n)\right|\right\} = \mathrm{MEDIAN}\left\{\left|y_{vi}(n) - \mathbf{C}_i\mathbf{x}(n \mid n-1)\right|\right\} \qquad (12\text{-}5)$$

Causal Detection

$$\mathrm{Corr}_{ij}(n) = \frac{1}{N_j} \sum_{k=-(N_j-1)}^{0} \mathrm{RS}_{ij}\left(N_j - 1 + k\right) Y_i(n+k) \qquad (12\text{-}6)$$

$$\mathrm{NCorr}_{ij}(n) = \left| \frac{\mathrm{Corr}_{ij}(n) - E\left[Y_i(n)\right] E\left[\mathrm{RS}_{ij}(n)\right]}{\sigma\left[Y_i(n)\right] \sigma\left[\mathrm{RS}_{ij}(n)\right]} \right| \qquad (12\text{-}7)$$

, where Reference Signals definition follows:

$$\mathrm{RS}_{ij}(k) = \begin{cases} e^{\frac{-k}{100}}, & j = 1, \quad k = 0,\ 1,\ \ldots,\ 500 \\ e^{\frac{-k}{300}}, & j = 2, \quad k = 0,\ 1,\ \ldots,\ 1000 \end{cases} \qquad (12\text{-}8)$$

Evaluation of Causal Detection

$$\mathrm{Enable}_{ij}(n) = \frac{\mathrm{Corr}_{ij}(n)}{\sigma\left[C_i x(n \mid n-1)\right]_j} \qquad (12\text{-}9)$$

Decision Making (Reasoning)

$$\begin{aligned} &\text{IF } \left\{\mathrm{NCorr}_{ij}(n) \geq 0.9\right\} \text{ AND } \left\{\mathrm{Enable}_{ij}(n) \geq 0.05\right\} \text{ THEN } \left\{\mathrm{BinaryCorr}_{ij}(n) = 1\right\} \\ &\text{ELSE } \left\{\mathrm{BinaryCorr}_{ij}(n) = 0\right\} \end{aligned} \qquad (12\text{-}10)$$

NCorr$_{ij}$(*n*) is input
BinaryCorr$_{ij}$(*n*) is output of this block

Pulse Prolonging Procedure for Preventing the Post-Exitation

$$\begin{aligned} &\text{IF } \left\{\mathrm{BinaryCorr}_{ij}(n) \text{ falled}\right\} \text{ THEN } \left\{\text{Prolong pulse by } N_j \text{ samples}\right\} \\ &\text{ELSE } \left\{\text{Clear or Hold On}\right\} \end{aligned} \qquad (12\text{-}11)$$

BinaryCorr$_{ij}$(*n*) is modified at the output of this block

Table 12.1-1 The timing diagram of fault algorithm inside of DKF$_i$ processor in an i-th node, where r$_i$(n) is a residual and C$_i$x(n|n-1) are inputs of FDI system; and BinaryCorr$_{ij}$(n) is one output.

The fault detection algorithm can be divided in two stages: residual generation and decision making.

1. In the first stage, the residual (shown in Figure 12.1-2 of blue curve and given by (12-4)), and the enhanced failure effect on the residual is called the fault signature (Figure 12.1-2 of red curve). Thus this effect should be utilised in fault detection algorithm. The residuals are obtained from observations using (12-4) in every node. As the example, after n = 1500 the appeared drift permanently gives exponentially increasing huge signal $r_1(n)$ depict in Figure 12.1-2. Then the absolute value $|r_1(n)|$ is smoothed by median filter and assigned to $Y_1(n)$, see (12-5) and [19].

Generally, median filter makes the smoothing as a low-pass filter. In every node, median filter takes 25 samples of $|r_i(n)|$ signal into an overlapping block. In digital signal processing this block is called a buffer of memory. Then these samples are reordered starting from their minimal value up to maximum value and stored into a new block of reordered samples. Only the 13-th sample of the new block is assigned to $Y_i(n)$.

2. In the second stage, the procedure of decision making is needed for fault detection to decide whether a fault is presented or not in residuals. A spurious fault is not our objective of study.

The fault signature $Y_1(n)$ is correlated with real drift $d_1(n)$ been presented in Figure 12.1-1. This is the eventual fact for fault detection. Here the problem is that real drift $d_1(n)$ is unknown for every processor in a node. Thus a presumption of possible drift profiles should be made from some previously done measurements or employed expert systems. Thus various drift models need to be created and stored in memory of each processor such as reference signals (12-8). Actually, it is difficult to exactly suppose any drift profile, so that the mathematical description of every reference signal may differ from real drift $d_1(n)$ but high correlation between them still exits to be suitable in fault detection.

The point is that, the permanently stored reference signals and measured $Y_1(n)$ are correlated by defining (12-6). Thus the cross-correlation (12-6) is high when a fault occurs because $d_1(n)$ is contained in $Y_1(n)$ and $y_{v1}(n)$. This procedure is performed in every node separately. Two reference signals $RS_{11}(k)$, $RS_{12}(k)$ are shown in Figure 12.1-3 and used in every node/processor.

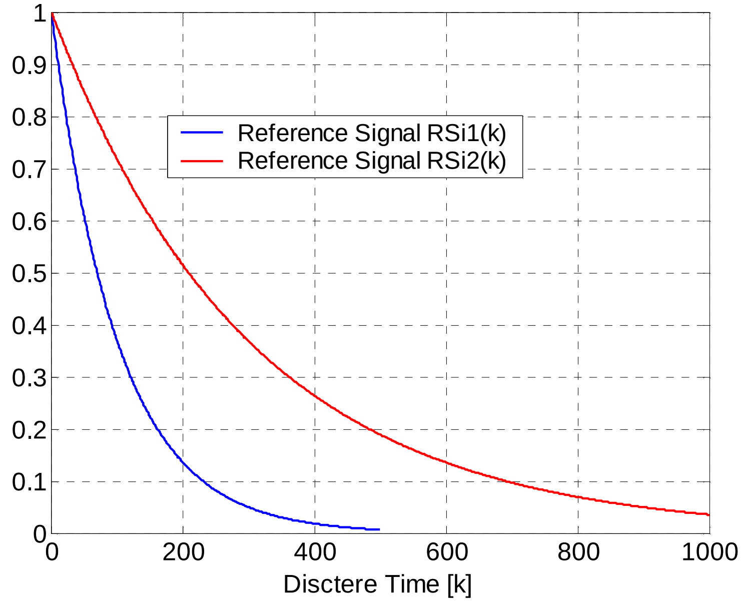


Figure 12.1-3 Reference signals $RS_{11}(k)$ and $RS_{12}(k)$.

Generally, every node may use different variety and volume of reference signals, but in our simulations only one set is used (12-8) such as $\{RS_{i1}(k), RS_{i2}(k)\}$. Of course, this fact is up to the real case. The subscription *i* refers to the existing *i*–th node. The mentioned correlation between measured $Y_1(n)$ of the first sensor and $RS_{11}(k)$, $RS_{12}(k)$ is presented in Figure 12.1-4 with the explained case of the first corrupted sensor by the exponential drift $d_1(n)$.

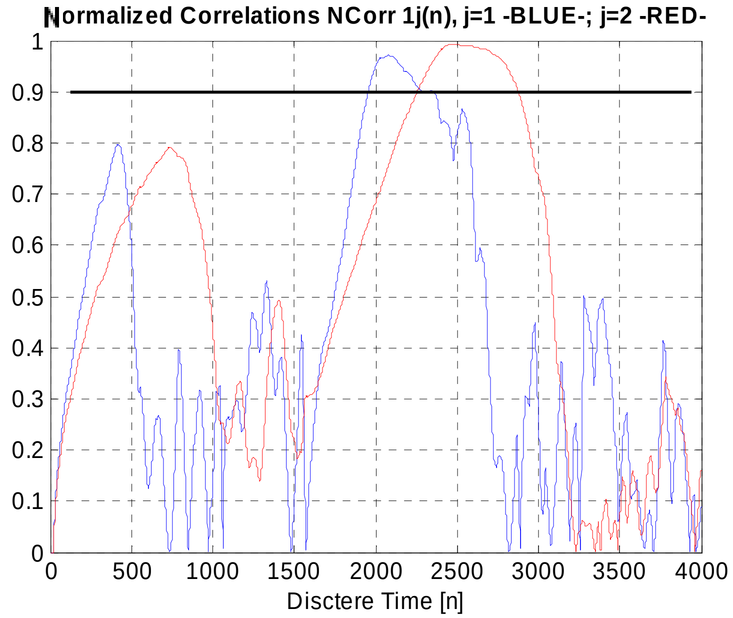


Figure 12.1-4 Computed correlation between a reference signals and measured $Y_1(n)$ inside of the first node.

The blue curve corresponds to correlation $NCorr_{11}(n)$ (computed by (12-7) and (12-6)) between $RS_{11}(k)$ and $Y_1(n)$. Finally, a fault caused by drift might be correctly detected only if some correlations $NCorr_{11}(n)$ and $NCorr_{12}(n)$ crossed threshold equal to 0.9 (solid black line) in conjunction with $Enable_{11}(n)$ and $Enable_{12}(n) \geq 0.05$. This decision making is given by (12-10). The $Enable_{11}(n)$ and $Enable_{12}(n)$ signals are found by (12-9) with $i = 1$ and $j = 1$ and 2 and both play an important role in the avoidance of spurious detections possibly caused by a very small power of residuals.

Then two bitwise pulses are generated by $BinaryCorr_{11}(n)$ and $BinaryCorr_{12}(n)$ in Boolean algebra. They are equal to one if valid fault has been detected, or zero if no fault has been detected. These two pulses are prolonged by 500 and 1000 periods and updated by (12-11) and shown in Figure 12.1-5.

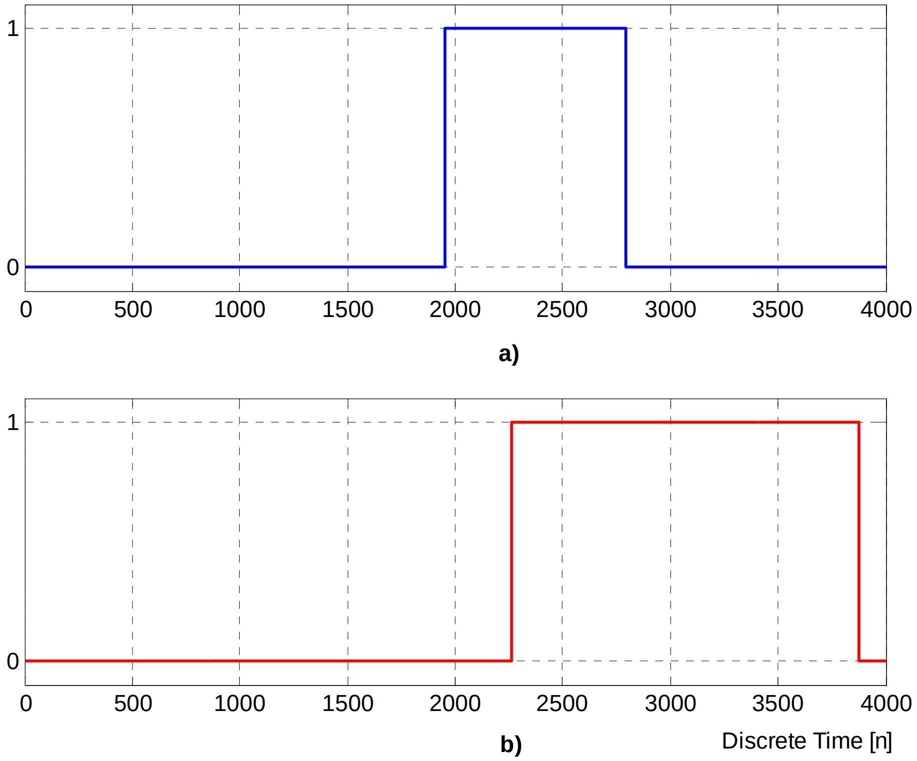


Figure 12.1-5 Two pulses after the fault detection:
a) BinaryCorr$_{11}$(n) pulse (fallen down at n = 2291) prolonged by 500 samples
b) BinaryCorr$_{12}$(n) pulse (fallen down at n = 2878) prolonged by 1000 samples.

Both signals $BinaryCorr_{11}(n)$ and $BinaryCorr_{12}(n)$ would be logically combined by performing bitwise logical OR operation to get a coupled pulse that starts at period of 1955 and ends at 3878. The process of fault detection has been described by the example of decentralize structure containing three sensors where the first of them accidentally failed.

Generally, every *i*-th node contains its processor with one sensor and a program of fault detection algorithm. In our case, after some measurements two reference signals are needed and therefore the process of fault detection is two times executed in real time after every received sample of $y_{vi}(n)$ from the plant. The Figure 12.1-6 briefly illustrates the fault detection method.

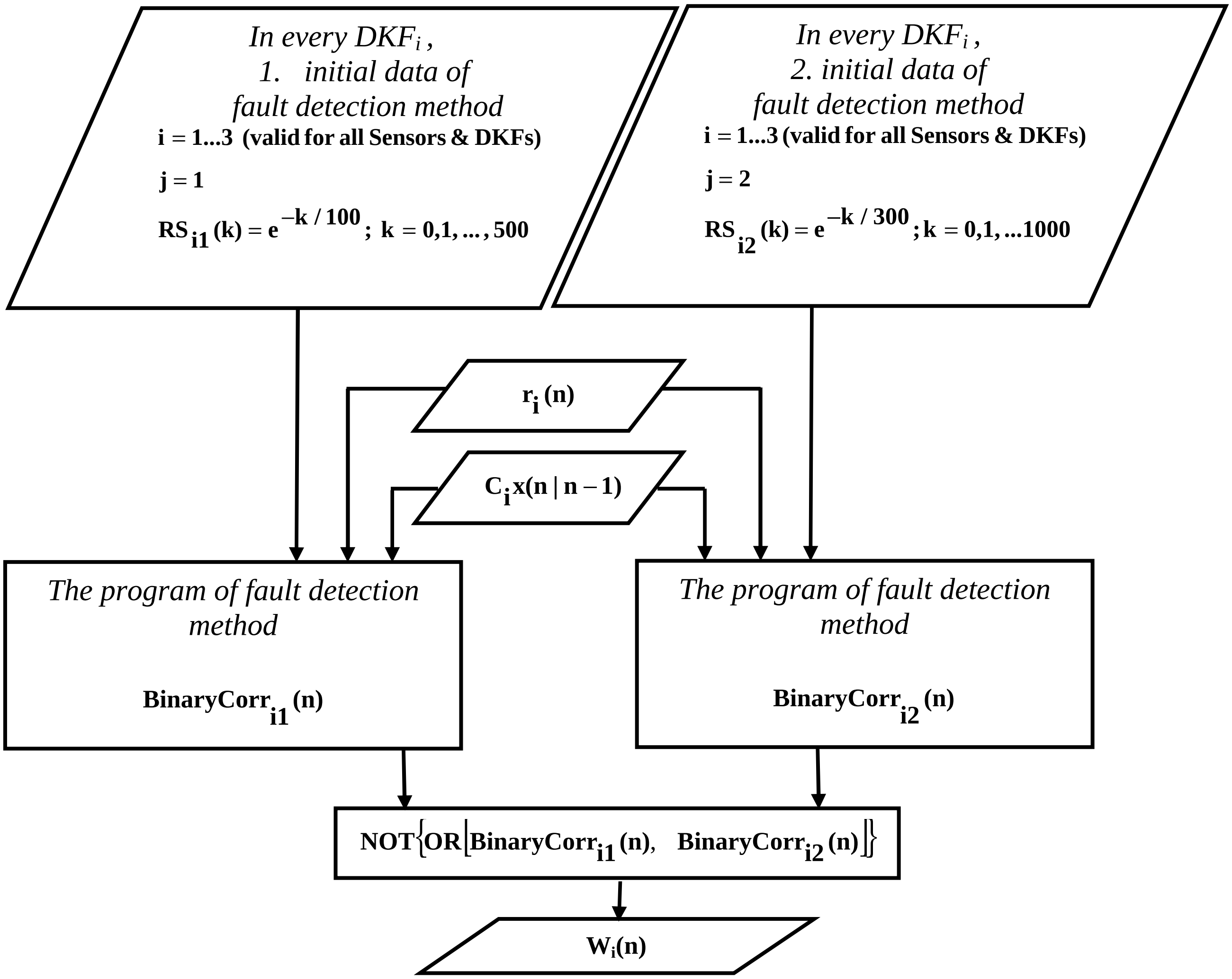

In every DKF_i ,
1. initial data of
fault detection method
i = 1...3 (valid for all Sensors & DKFs)
j = 1
$RS_{i1}(k) = e^{-k/100}$; k = 0,1, ..., 500
In every DKF_i ,
2. initial data of
fault detection method
i = 1...3 (valid for all Sensors & DKFs)
j = 2
$RS_{i2}(k) = e^{-k/300}$; k = 0,1, ...1000
$r_i(n)$
$C_i x(n | n-1)$
The program of fault detection method
$BinaryCorr_{i1}(n)$
The program of fault detection method
$BinaryCorr_{i2}(n)$
$NOT\{OR[BinaryCorr_{i1}(n), BinaryCorr_{i2}(n)]\}$
$W_i(n)$

Figure 12.1-6 Flowchart of fault detection (FD) method.

The *i*-th processor receives observations $y_{vi}(n)$ over the *i*-th sensor, optimally estimates state vector $\boldsymbol{x}(n|n)$ by DKF (Unit 4) and executes the program of FD method. To obtain a needed response while the fault has been detected, the FD needs three inputs such as residuals $r_i(n)$, priori filtered output $\boldsymbol{C}_i\boldsymbol{x}(n|n\text{-}1)$. Then a priori reference signals $RS_{i1}(k)$, $RS_{i2}(k)$ are needed too. The program of FD method responses an either pulse $BinaryCorr_{i1}(n)$ or $BinaryCorr_{i2}(n)$ that indicates the detected sensor fault while equal to Boolean-One. When this pulse is to be fallen down to zero, it is continuing prolonged at for next 500 or 1000 periods of time depending on $RS_{i1}(k)$, $RS_{i2}(k)$, respectively. Finally

BinaryCorr$_{i1}$(n) and *BinaryCorr*$_{i2}$(n) are logically combined by OR to get a coupled pulse been one's complemented (negated) after to get the weighting factor $W_i(n)$. The $W_i(n)$ are common members between presented fault detection and fault isolation method.

Fault isolation. This part presents the FI algorithm to the important theme of sensor validation in decentralized structure in multi-sensor fusion. Of course generally, a major number of sensors can be affected by a drift at the same time.

The main focus on fault isolation is based on variance error information $\boldsymbol{VEI}_i(n)$ and state error information $\boldsymbol{SEI}_i(n)$ validation. These two matrices are usually corrupted by a fault while the ***i***-th sensor is affected by a drift. During successful fault detection, the corrupted $\boldsymbol{VEI}_i(n)$ and $\boldsymbol{SEI}_i(n)$ matrices would be isolated only by the node wherein its corresponding sensor has been found incorrect by FD program implemented inside of a corresponding node. The $W_i(n)$ results two instances as

$$W_i(n) = \begin{cases} 0, \text{when fault detected} \\ 1, \text{when no fault detected} \end{cases} . \tag{12-12}$$

In DKF estimator of every node the $\boldsymbol{VEI}_i(n)$ and $\boldsymbol{SEI}_i(n)$ are replaced by the matrix multiplications

$$W_i(n)\boldsymbol{VEI}_i(n) \ , \tag{12-13}$$

and

$$W_i(n)\boldsymbol{SEI}_i(n) \tag{12-14}$$

, respectively.

This procedure will be shown in Figure 12.1-7. In other words, this weighting factor $W_i(n)$ is found to cross the $\boldsymbol{VEI}_i(n)$ and $\boldsymbol{SEI}_i(n)$ matrices using (12-13) and (12-14).

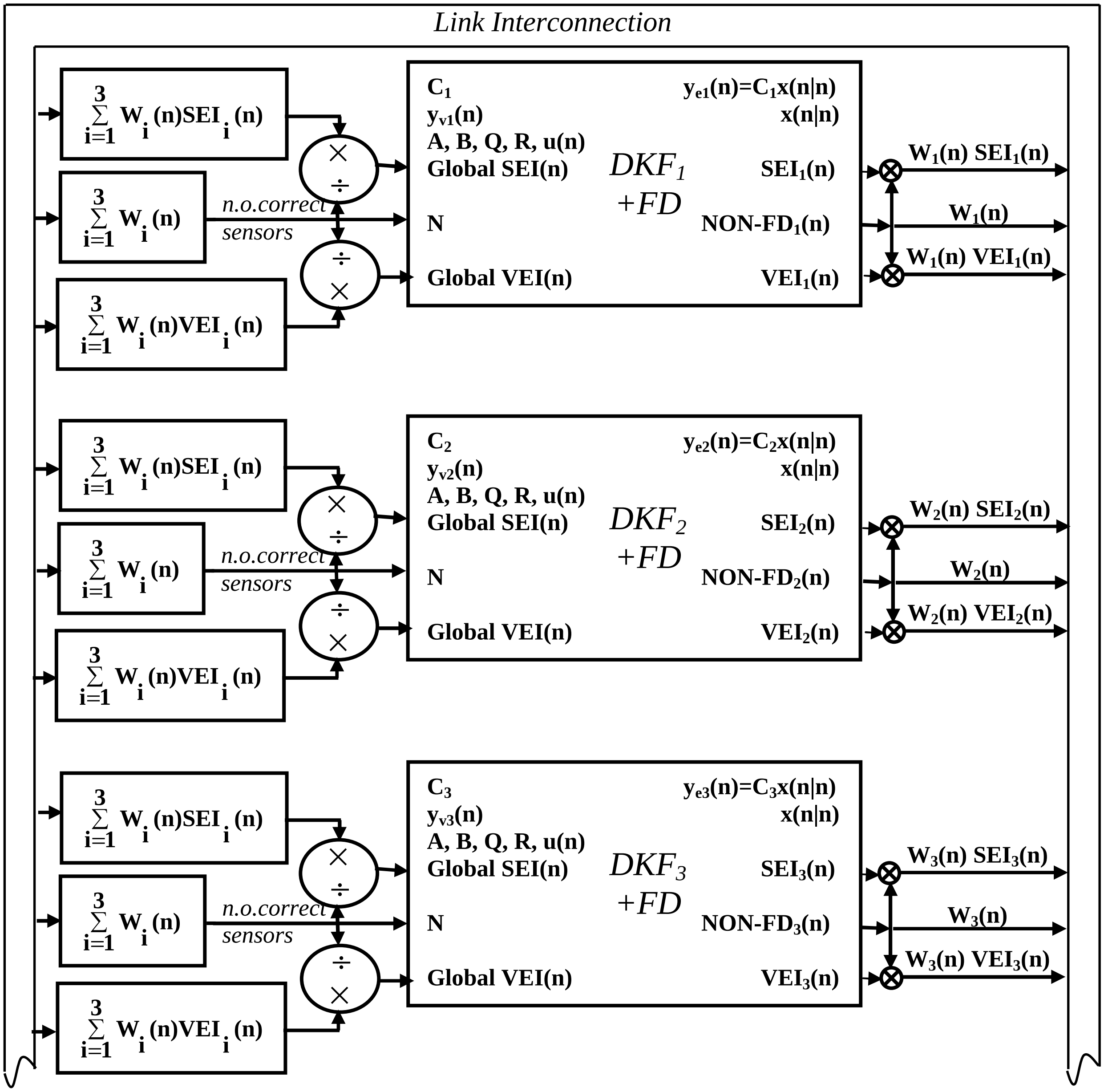


Figure 12.1-7 DKF with fault detection system inside and isolation system around.

Every block of processor is denoted by DKF_i+*FD*. An outage part of Figure 12.1-7 performs the fault isolation according to (12-13) and (12-14). Thus the FI crosses matrices $\boldsymbol{VEI}_i(n)$, $\boldsymbol{SEI}_i(n)$ via the weighting factor $W_i(n)$ when necessary. Then a number of correctly performing sensors is lower than the total number of sensors if a failure occurs.

Hence the number of correctly performing sensors can be computed as follows:

$$N = \sum_{i=1}^{3} W_i(n), \tag{12-15}$$

where we have three sensors totally in our simulations. These data of (12-13) – (12-15) are broadcasted through the link interconnection and known at receivers part of every node in Figure 12.1-7. Finally, equations (4-10) and (4-11) are employed with whole fusion, state estimation, and correct fault isolation. The remaining inputs and outputs presented in Figure 12.1-7 are used only in estimation. The executed FDI algorithm has some shown test results below.

Testing the DKF, which contains low power level of process noise and one failed sensor in decentralized structure of multi-sensor fusion. The FDI performance of weights from $W_1(n)$ to $W_3(n)$ and the number of correct sensors N are demonstrated in Figure 12.1-8 below.

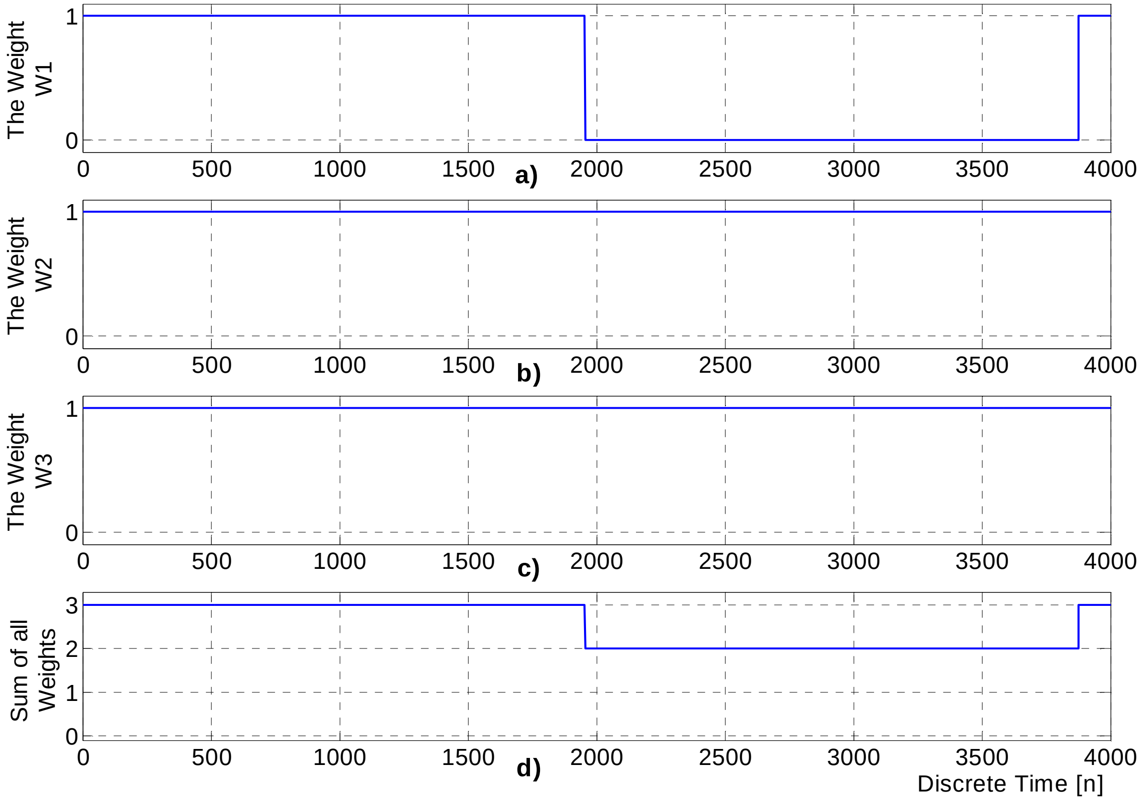


Figure 12.1-8 The performance of fault detection system depicted by:

a) plotted weight $W_1(n)$
b) plotted weight $W_2(n)$
c) plotted weight $W_3(n)$
d) the sum of all weights equal to the number of correct sensors N.

This example shows the functionality of estimator with FDI method containing three sensors in DKF structure. Low power level of process noise (6-3) is used in this test, and first sensor fails starting at *n* = 1500 and finishing by *n* = 4000. The first node detects the fault of its own sensor with the FDI method during the detection term when *n* = 1955 … 3878, see Figure 12.1-8. The isolation of the sensors fault is expected immediately when it starts, but our FDI method needs about 500 periods to decide if the fault has occurred similar to drift profile (compared to reference signals (12-8) in processors memory). This reliability of system detection refers to correlation function performance with corresponding products $NCorr_{11}(n)$, $NCorr_{12}(n)$, threshold equal to 0.9, see (12-10), and prolonged pulses $BinaryCorr_{11}(n)$, $BinaryCorr_{12}(n)$ produced by the first node/sensor. The next two weights remain unchanged due to second and third sensors re correct.

In all sections of FDI tests, we like to specify the related MSE of state estimation. At first, it is needed to remark on the state vector of plant $\boldsymbol{x}(n)$ and the estimated state vector $\boldsymbol{x}(n|n)$ of the model with FDI method seen in Figure 12.1-7. Their difference $\boldsymbol{a}(n)$ is given by (7-1), and the mean square error, MSE, of the state estimation is computed using (6-6) formula. The accuracy of state estimation is measured by MSE in proportion to power of plant state, called relative MSE. This power is given by:

$$\frac{1}{4001}\sum_{n=0}^{4000}\boldsymbol{x}^{T}(n)\boldsymbol{x}(n), \tag{12-16}$$

where $n = 0,\ 1,\ ...,\ 4000$ during MATLAB simulation. This relative 99.96% MSE is measured when all sensors have worked without any fault at $n = 0,\ 1,\ ...,\ 1500$; and it is presented in Table 12.1-2.

The accuracy of state estimation in the structure of DKF during <0; 1500>	The reliability of FDI in the structure of DKF during <1501; 4000>
$9.996 \,.\, 10^{-1}$	$7.460 \,.\, 10^{-1}$

Table 12.1-2 Related MSE of state estimation, and the reliability of FDI technique.

A new criteria needs to be specified to evaluate a reliability of parallel DKF structure with FDI. At the periods $n = 1501,\ 1502,\ ...,\ 4000$ during the failure, the reliability is found:

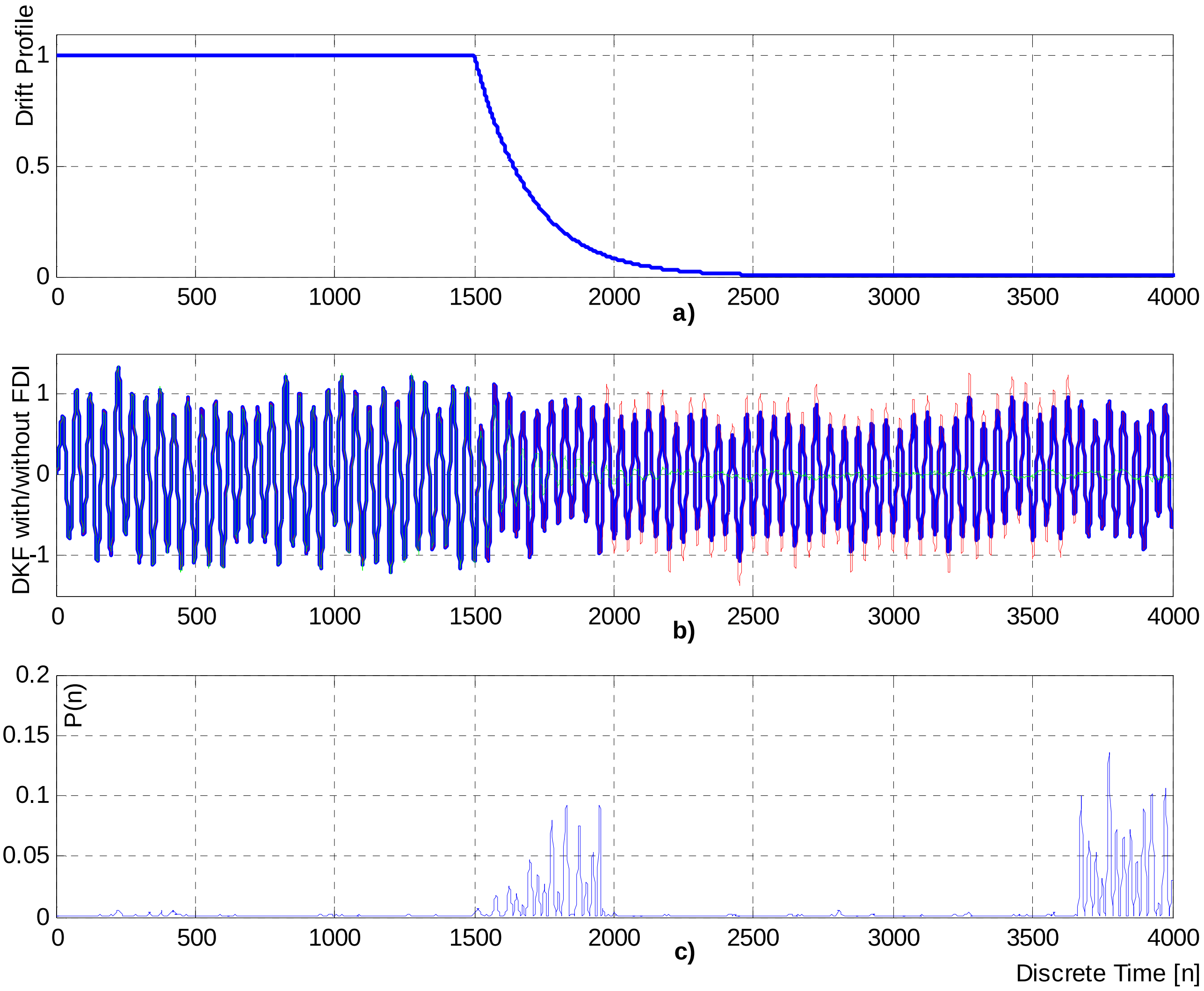
$$1-\frac{MSE \mid \boldsymbol{x}(n)-\boldsymbol{x}_{FDI}(n \mid n)}{MSE \mid \boldsymbol{x}(n)-\boldsymbol{x}(n \mid n)}. \tag{12-17}$$

The $\boldsymbol{x}_{FDI}(n \mid n)$ is a state vector managed by the Figure 12.1-7 and $\boldsymbol{x}(n \mid n)$ is a fully corrupted state vector without FDI. Thus $\boldsymbol{a}(n)$ is conditionally defined in fraction of (12-17). This fraction determines how much energy needs to compensated in states more, but there is nothing to do with state estimation accuracy. This test gives about 75% FDI reliability in Table 12.1-2.

Testing the DKF, which contains high power level of process noise and one failed sensor in decentralized structure of multi-sensor fusion. In this part, we demonstrate the performance of previous system with 100-times higher power level of process noise. Here the reliability is about 70% and almost 5% lower than the obtained one from the previous case of a little process noise, see Table 12.1-3. Illustrated signals are additionally shown in Figure 12.1-9.

The accuracy of state estimation	The reliability of FDI
in the structure of DKF	in the structure of DKF
during <0; 1500>	during <1501; 4000>
$9.990 \,.\, 10^{-1}$	$6.966 \,.\, 10^{-1}$

Table 12.1-3 Related MSE of state estimation, and the reliability of FDI technique.



a)

b)

c)

Figure 12.1-9 Plotted signals of DKF system with FDI conditionally with low power level of process noise:

a) the profile of drift at the first sensor,

b) the observation $y_{v1}(n)$ of the first sensor in GREEN; filtered output $y_{e1}(n)$ of DKF with FDI in RED; signal of pure filtered output of DKF without any FDI plotted by BLUE color,

c) state error variance measured at DKF estimation of the system with FDI technique.

The FDI detects the failure at periods between 1953 and 3652, and consequently isolates the sensor fault however the drift occurs between 1501 and 4000 as known in Unit 12. The state estimation accuracy is high when no drift appears.

The first Figure 12.1-9a demonstrates profile of the exponential drift mentioned above and known such as the sensors fault. There we can see the red curve of compensated power done by the FDI method. The Figure 12.1-9c demonstrates the state error variance $var[\boldsymbol{a}(n)]$, where $\boldsymbol{a}(n) = x(n) - x_{FDI}(n \mid n)$. The variance is highly produced during the term about the times 1500 to 2000, when the fault detection is in the process and the state estimation is corrupted. Also a high error is produced at the state estimation from the time 3653 until the end of simulation due to unregonisable drift in residuals.

12.2 Testing the FDI implemented in DKF structure with two failed sensors

This section deals with the same example as previous one, but the first two sensors are affected by two different drifts and one sensor remains correctly doing. The observations model is defined as:

$$\begin{bmatrix} y_{v1}(n) \\ y_{v2}(n) \\ y_{v3}(n) \end{bmatrix} = \begin{bmatrix} d_1(n) & 0 & 0 \\ 0 & d(n) & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}(n) + \begin{bmatrix} v_1(n) \\ v_2(n) \\ v_3(n) \end{bmatrix}, \qquad (12\text{-}18)$$

where the time $n = 0, \ 1, \ \ldots, \ 4000$ remains unchanged in simulation. The drift $d_1(n)$ is taken over the Section 12.1, and the slow $d(n)$ is defined by (11-2) with $\delta = 1500$ and from the Unit 11. The powers of sensor noises are defined by (6-4). The time behaviours of the drifts are shown in below.

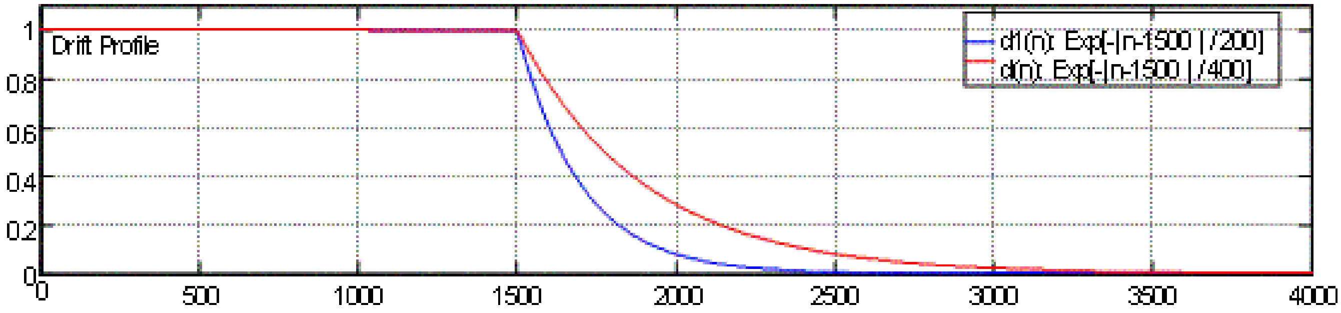


Figure 12.2-1 Drifts d_1(n) and d(n).

Testing the DKF, which contains low power level of process noise and two failed sensors in decentralized structure of multi-sensor fusion. The state estimation accuracy remains still the same when no failure has been caused by sensors at $n = 0,\ 1,\ \ldots,\ 1500$. Although the reliability of FDI method is about 14% higher than presented one in Section 12.1, because signals of first two sensors are not the same quality. These facts refer to table below.

The accuracy of state estimation	The reliability of FDI
in the structure of DKF	in the structure of DKF
during <0; 1500>	during <1501; 4000>
$9.996 \, . \, 10^{-1}$	$8.894 \, . \, 10^{-1}$

Table 12.2-1 Related MSE of state estimation, and the reliability of FDI technique.

The FDI performance in weights $W_1(n)$ to $W_3(n)$ and the number of correct sensors N are demonstrated in Figure 12.2-2. There the weights $W_1(n)$ and $W_2(n)$ fall down when sensors faults have been detected. The third remaining sensor is not corrupted, which fact corresponds to the weight $W_3(n)$ that ensures fully transmitted $\boldsymbol{VEI}_3(n)$ and $\boldsymbol{SEI}_3(n)$ over the link interconnection. The Figure 12.2-2c illustrates the number of correct sensors obtained by the applied sum (12-15) of $W_1(n)$ to $W_3(n)$.

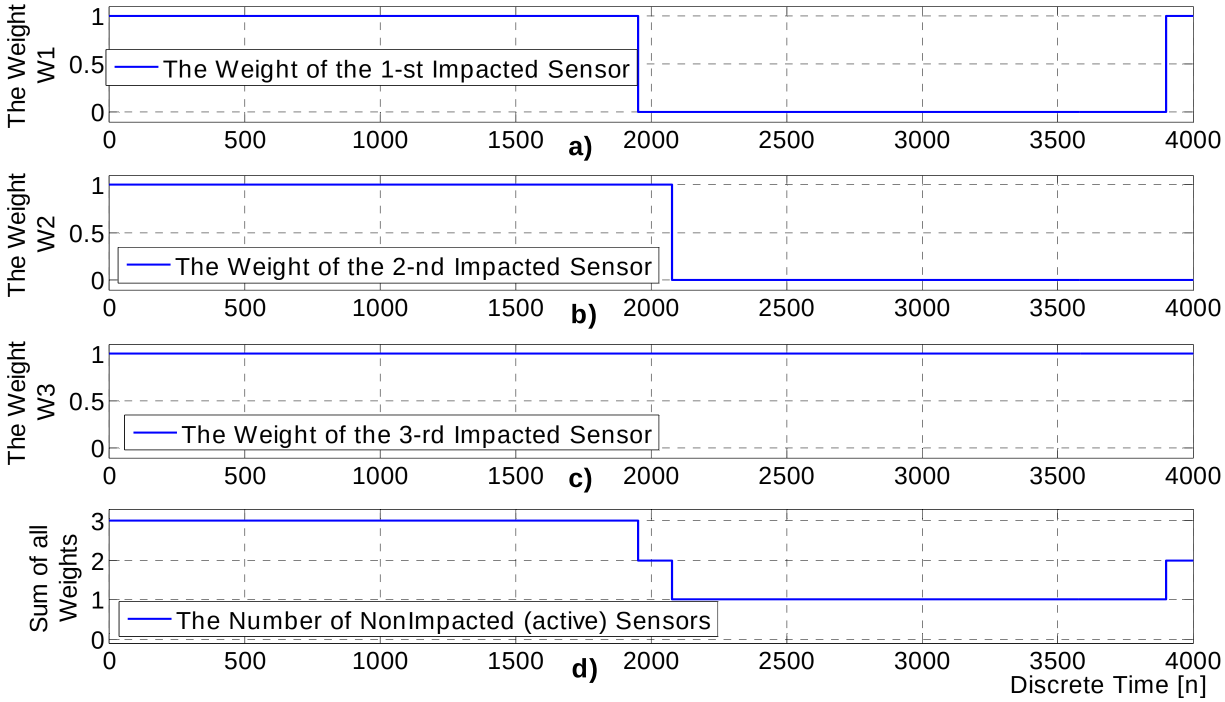


Figure 12.2-2 The performance of fault detection system depicted by:

e) plotted weight $W_1(n)$
f) plotted weight $W_2(n)$
g) plotted weight $W_3(n)$
h) the sum of all weights equal to the number of correct sensors N.

Testing the DKF, which contains high power level of process noise and two failed sensors in decentralized structure of multi-sensor fusion. Here the FDI reliability is smallest of all, abut only 54%, and shown below in table. High power of process noise affects the functionality of FDI system. This fact is encountered with spurious detection of fault in seldom fallen weight $W_3(n)$. This is the imperfection of FDI simply basing on cross-correlation function. One can say that this is the limit that the FDI algorithm refuses to work and the DKF is not under control.

The accuracy of state estimation in the structure of DKF during <0; 1500>	The reliability of FDI in the structure of DKF during <1501; 4000>
$9.990 \cdot 10^{-1}$	$5.388 \cdot 10^{-1}$

Table 12.2-2 Related MSE of state estimation, and the reliability of FDI technique, with high power of w(n).

In Figure 12.2-3, we can see two short periods (2100-2650 and 3550-4000 approximately) when the DKF works without any control and $\boldsymbol{Q}(n)$ and $R(n)$are substituted by some small defaulted values.

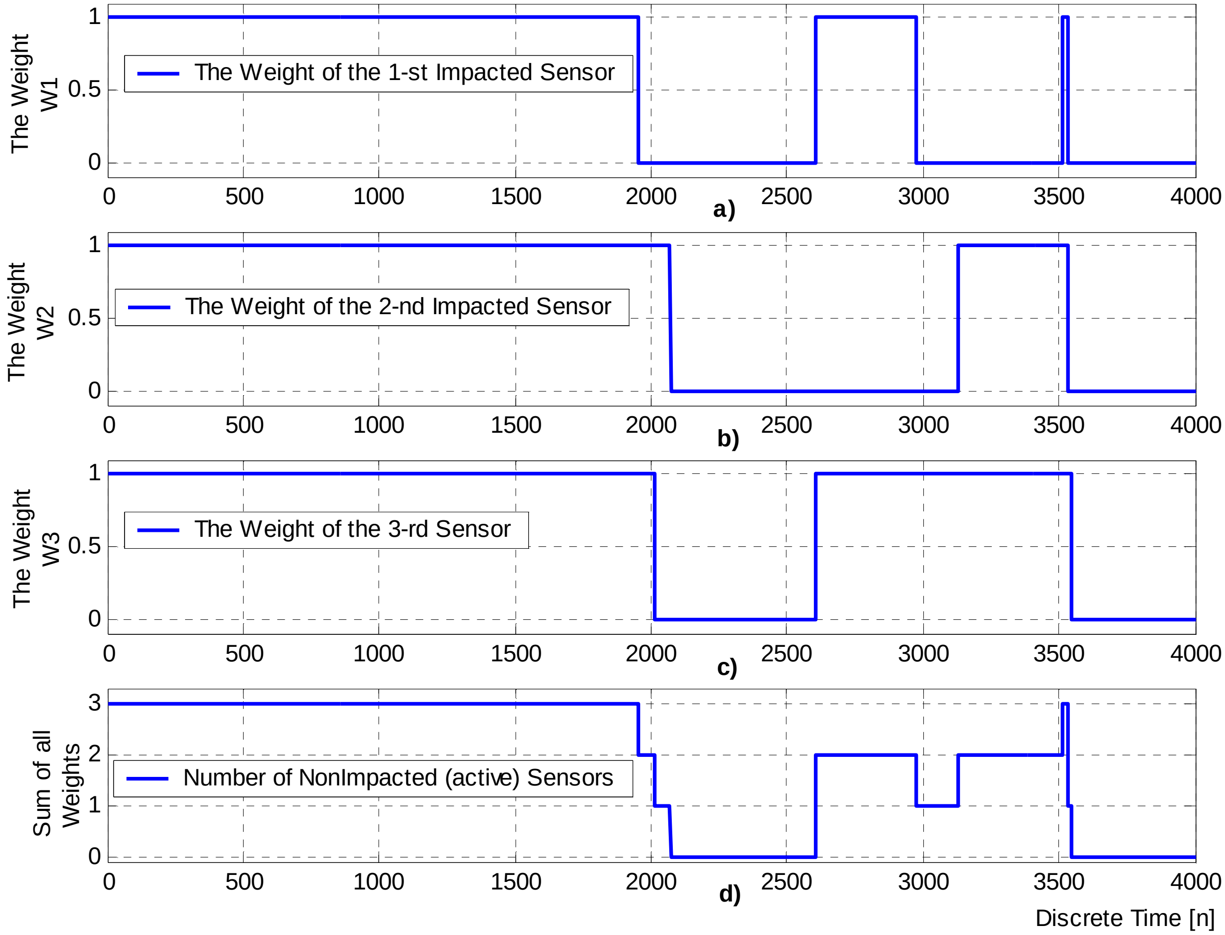


Figure 12.2-3 The DKF system with FDI and its functionality under high power of process noise:

i) plotted weight $W_1(n)$
j) plotted weight $W_2(n)$
k) plotted weight $W_3(n)$
l) the sum of all weights equal to the number of correct sensors N.

12.3 Conclusions

Below we apply the fifth question of section 1.1 to this conclusion to be answered:
"Find out a fault detection and isolation, FDI, algorithm been possibly useable for DKF model. This model has to be tested with a minor number of affected sensors by exponentially descending drifts. What is a performance of this DKF model with applied FDI algorithm, when:

- one sensor is affected by the drift, and additionally two sensors are correct;
- two sensors are affected by two miscellaneous drifts, plus one correct sensor?"

The major requirement on this fault detection and isolation, FDI, algorithm is an availability to multiple observations. We have shown on how to apply FDI algorithm to the problem of faults in sensors. The FDI algorithm has been found out in three strategies:

1. exercising on fault detection method, FD
2. exercising on fault isolation method, FI.

The all algorithm is tailor-made on system dynamics and the knowledge of drift in sensors.

Firstly the question related to one affected sensor will be answered below. Our simulation results show that the DKF model with FDI algorithm yields 75% reliability of FDI performance insomuch that a pure DKF model without FDI provides exactly 0% reliability. The FDI reliability is not meant to be state estimation accuracy, although its relative MSE is about 0.01. In case of high process noise, results of simulations show about 70% reliability of FDI algorithm. The five percentage aggravation is caused by 100-times higher power of process noise, where the cross-correlation function used in FDI system is disturbed by that.

Having two sensors affected by exponential drifts, the FDI reliability rises up to 84% which describes the benefit of FDI. But considering 100-times higher power of process noise, the reliability is reduced down to 54%, and relative MSE of state estimation is about 0.11. This is the FDI imperfection with major number of corrupted sensors. Two failed sensors out of three total sensors and having high power of process noise result in loss of performance of FDI. So said, the FDI functionality is correct as long as a number of affected sensors is minor otherwise the DKF may be at risk.

The reliability of FDI describes on how much lost energy in $x(n \mid n)$ is compensated back rather than the pure DKF without FDI can not perform.

The answer of fault detection and isolation algorithm has been recently given above.

Simulation conditions:

The DKF models use multi-state; multi-sensor system fusion with different quality of observations. Time behaviour of $u(n)$ is sinus as a still oscillated signal rather used than a less problematic step pulse.

13 ADAPTIVE FUZZY LOGIC KALMAN FILTER ALGORITHM AND EXERCISES

A fuzzy based approach to adaptive algorithm is developed for AKF's enhancements to Kalman Filter, KF, covering practical needs. We focus on a significant difficulty in linear KF to trace a priori knowledge of the process covariance matrix $\boldsymbol{Q}(n)$ and observation noise variance $\boldsymbol{R}(n)$. In most practical applications of manoeuvring or vehicle navigation, the $\boldsymbol{Q}(n)$ and $\boldsymbol{R}(n)$ are unknown or even initially estimated. The problem here is that the optimality of the estimation in the KF is closely connected to the quality of a priori information about the process and observation noise.

Insufficiently known priori filter statistics, $\left(\boldsymbol{x}_0, \boldsymbol{Q}, \boldsymbol{R}\right)$, can either reduce the precision of state estimation or introduces biases on the other hand, see Unit 9. Additionally, incorrect priori information leads to practical divergence of the filter [20].

This unit deals additionally with an innovation adaptive estimation, IAE, approach coupled with fuzzy logic methodology used to adjust the $\boldsymbol{Q}_{\boldsymbol{a}}(n)$ diagonal matrix for the adaptive Kalman filter - AKF. The use of fuzzy-rule based adaptation algorithm will be explored to cope with priori $\boldsymbol{R}(n)$ and $\boldsymbol{Q}(n)$ matrices. A choice to fuzzy membership functions for the adaptation technique is carried out with heuristic approach. The membership functions for the IAE approach are established by the combination of cross-correlation functions and an experience.

In [18], fine-tuning of genetic algorithms, GAs, has the ability to find membership functions performance closer to optimal solutions. Despite, the approach to adaptation without any GAs would be presented in next sections. Main idea of fuzzy KF is taken over [18]. Brief introduction to single-sensor adaptive Kalman filtering has been given.

First of all, two types of system would be applied to estimation. The first proposed system can be described by (12-1) and (12-2) such as a third order filter. Fully described model can by found in Unit 12. The second system follows the physical model of a plant described as the tenth order filter's behaviour, where $\boldsymbol{A}$ matrix, $\boldsymbol{B}$ and $\boldsymbol{C}_i$ vectors are given by (13-1) to (13-3) respectively.

$$\begin{bmatrix} 7.107 & 8.326.10^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -27.38 & 0 & 8.326.10^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 62.678 & 0 & 0 & 8.326.10^{-1} & 0 & 0 & 0 & 0 & 0 & 0 \\ -94.407 & 0 & 0 & 0 & 8.326.10^{-1} & 0 & 0 & 0 & 0 & 0 \\ 97.75 & 0 & 0 & 0 & 0 & 8.326.10^{-1} & 0 & 0 & 0 & 0 \\ -70.452 & 0 & 0 & 0 & 0 & 0 & 8.326.10^{-1} & 0 & 0 & 0 \\ 34.898 & 0 & 0 & 0 & 0 & 0 & 0 & 8.326.10^{-1} & 0 & 0 \\ -11.369 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8.326.10^{-1} & 0 \\ 2.199 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8.326.10^{-1} \\ -1.918.10^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{13-1}$$

$$\begin{bmatrix} 1.52.10^{-2} & -9.25.10^{-2} & 2.303.10^{-1} & -2.671.10^{-1} & 4.64.10^{-2} & 2.833.10^{-1} & -4.08.10^{-1} & 2.753.10^{-1} & -9.76.10^{-2} & 1.47.10^{-2} \end{bmatrix}^{\mathrm{T}} \tag{13-2}$$

$$\begin{bmatrix} \boldsymbol{C}_1 \\ \boldsymbol{C}_2 \\ \boldsymbol{C}_3 \\ \boldsymbol{C}_4 \\ \boldsymbol{C}_5 \\ \boldsymbol{C}_6 \\ \boldsymbol{C}_7 \\ \boldsymbol{C}_8 \\ \boldsymbol{C}_9 \\ \boldsymbol{C}_{10} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{13-3}$$

In the addition to the tenth order filter, the system dynamics can be found by writing:

$$\boldsymbol{x}(n+1) = \boldsymbol{A}\boldsymbol{x}(n) + \boldsymbol{B}[u(n) + w(n)], \quad \boldsymbol{x}(0) = \mathbf{0}; \tag{13-4}$$

$$[y_{vi}(n)] = [\boldsymbol{C}_i \boldsymbol{x}(n)] + [v_i(n)], \tag{13-5}$$

where $v_i(n)$ are uncorrelated regarding to their variances proposed in Unit 6, and the subscript $i = 1, \ldots, 10$ corresponds to the i-th node. We have two alternatives of power level of $w(n)$, see Unit 6. In general way of both types of the system, all the noises (process and sensor) are uncorrelated being the important note here.

A control signal is commonly used for both systems:

$$u(n)=\begin{cases}0 \quad \text{when} \quad n=0,\ \ 1,\ \ ...,\ \ 100; \\ 1 \quad \text{when} \quad n=101,\ \ 102,\ \ ...,\ \ 300\end{cases} \tag{13-6}$$

Both systems are comparable in a term of power spectral density, PSD, processed from first element $\boldsymbol{x}(1,n)$ of $\boldsymbol{x}(n)$ providing corresponded quality. Their common PSD is shown in Figure 13-1. The reason to make experiments on two systems of the same quality is to show good functionality of AKF. In addition to that fact, the state estimation accuracy depends on a system sensitivity due to a delta of $\boldsymbol{Q_a}(n)$ and $R_{ai}(n)$ scalar as much as high order of the system is employed, see [3].

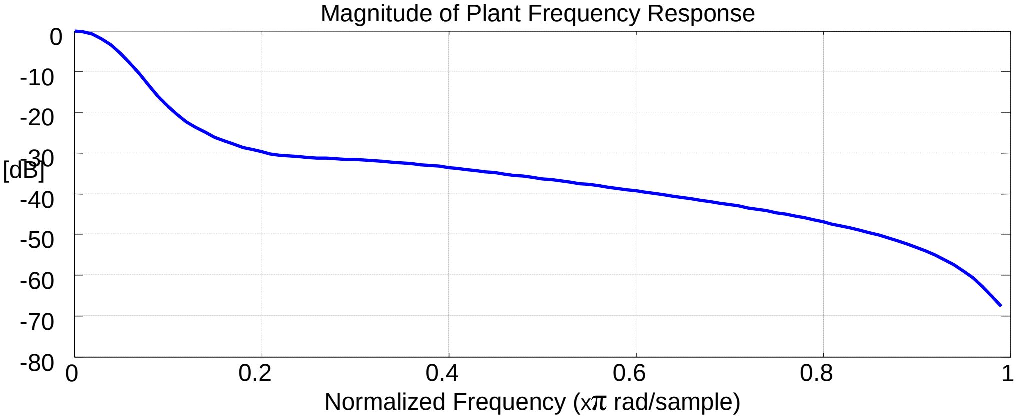


Figure 13-1 Magnitude of systems frequency response.

As one can see in Figure 13-1, the sampling frequency is much higher than corner frequency belonged to dominant bandwidth. There are several reasons for high sampling frequency [2]:

- The first one is that practically it might be difficult to determine a corner frequency due to attenuation of higher unwanted spectral bandwidth as well as anti-aliased filtering.
- The real issue in selection of a sampling frequency is only not a bandwidth quality but its uncertainty. If sampling at Nyquist frequency was reduced only about two samples per each cycle of oscillation, which would be too low frequency for practical reason. Thus sinusoidal signal behaves unstable phase response near to Nyquist frequency to be insufficient fact in control. On the other hand, if we use a sampling frequency that is much higher than resonant frequency, it would be reasonable to suppose that slight changes in the resonant frequency would have minimal effect in dumped phase response.

The following experiments will explore more on adaptive fuzzy logic Kalman filter been shortly abbreviated with AKF. In general way of the sensor noise $v_1(n)$, its covariance $\boldsymbol{R}(n)$ is a matrix, but all KF models allow sensor noise variance $R(n)$ to be scalar and adapted $R_a(n)$ are in AKF.

At first, some experiments would precede the adaptive algorithm to get us to know what items should be adjusted. The basic idea of this algorithm is to make the $\boldsymbol{Q_a}(n)$, $R_a(n)$ approaching to their theoretical values $\boldsymbol{Q}(n)$ and $R(n)$. This unit is coupled with Unit 5 in which three models are presented with missing algorithm providing $\boldsymbol{Q_a}(n)$. This algorithm is to be investigated.

Testing on how the innovation KF gain M(n) behaves. Let us perform following test

$$\boldsymbol{Q}\ (n)=\begin{cases}10^{-1}\mathbf{I}, & at \quad n=0,\ 1,\ ...,\ 20\\ \mathbf{I}, & at \quad n=0,\ 1,\ ...,\ 70\end{cases}$$
$$R_1(n)=\begin{cases}10^{-3}, & at \quad n=0,\ 1,\ ...,\ 20\\ 10^{-2}, & at \quad n=0,\ 1,\ ...,\ 70\end{cases} \tag{13-7}$$

describing excitation step pulses $R(n)$ and $\boldsymbol{Q}(n)$ into the second stage of Kalman filter, wherein the matrix $\boldsymbol{P}(n|n)$ and consequent $\boldsymbol{M}(n)$ are computed. In Figure 13-2, the filter time behaviour of $\boldsymbol{M}(n)$ vector elements produces some transients after n = 20, and slightly reaches stable value.

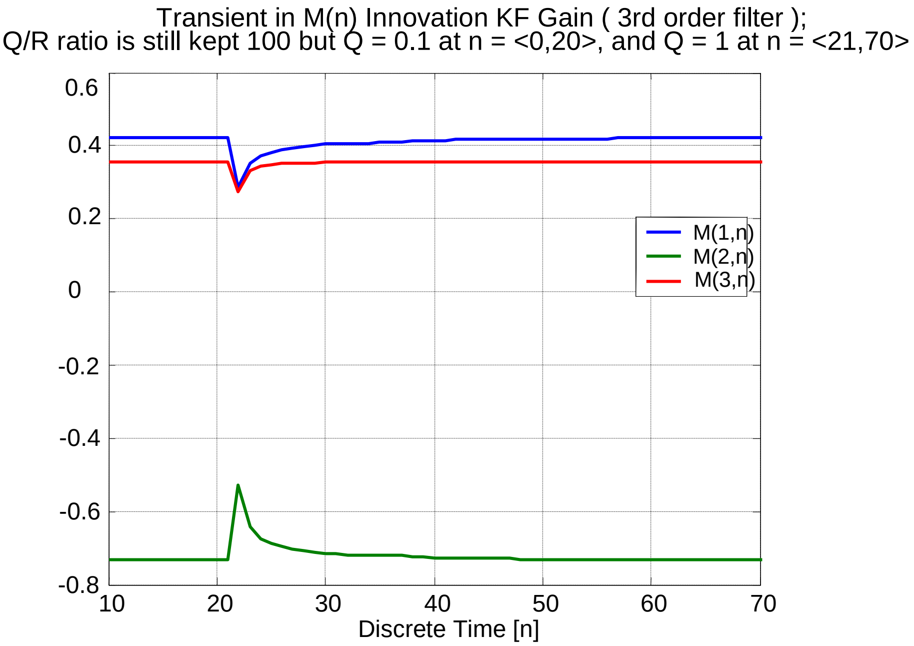


Figure 13-2 Test of innovation KF gain.

Next we like to obtain functional dependence between stable $\boldsymbol{Q}$ to $R(n)$ ratio and stable $\boldsymbol{M}(n)$ vector as an approximation, which might be done in a short range defining formula [21] :

$$\boldsymbol{M} \approx \left[\sqrt{\frac{Q}{R_1} + 1} - 1 \right] \boldsymbol{M_0}\,, \tag{13-8}$$

where $\boldsymbol{Q}$ is diagonal covariance matrix of process noise, and one of its equipollent elements from diagonal is denoted by Q as a scalar. In this approximation approach the vector $\boldsymbol{M_0}$ is reference gain corresponded to Q to R ratio equal to 1/1. An approximation result is shown in Table 13-1.

Q / R_1	**M**	**Approximated M**
0.1 / 1	[5.71 10^{-3}; -9.531 10^{-3}; 4.135 10^{-3}]	[5.032 10^{-3}; -8.355 10^{-3}; 3.629 10^{-3}]
1 / 1	[4.27 10^{-2}; -7.09 10^{-2} ; 3.08 10^{-2}]	≡ *Reference Gain*
10 / 1	[1.65 10^{-1}; -2.739 10^{-1}; 1.223 10^{-1}]	[2.388 10^{-1}; -3.965 10^{-1}; 1.723 10^{-1}]

Table 13-1 Approximated innovation gain as a vector.

To conclude this test, the $\boldsymbol{M}(n)$ can be approximated as a steady vector but the transients still need to be a function of $\boldsymbol{P}(n|n)$ and its time-update. This experiment emphasizes vast functionality of $\boldsymbol{M}(n)$.

How to adapt the gain M(n) and the estimation method. This part presents algorithm to the important topic. From the prior test the following circumstances are stated:

1. First event, that the $\boldsymbol{M}(n)$ is a function of both uncertainty $Q(n)$ and $R(n)$ parameters and their ratio (as the Q-to-R ratio) determines the steady value of $\boldsymbol{M}(n)$ vector by using the approximation formula (13-8) among near points. Thus, the true setting of those parameters is well founded and considerable in optimal state estimation.
2. Second event, $Q(n)$ is directly proportional to $\boldsymbol{M}(n)$ as to be steady.
3. Third event, R is inversely proportional to $\boldsymbol{M}(n)$ as to be steady.
4. Fourth event, that the transients of $\boldsymbol{M}(n)$, see Figure 13-2, can not be approximated in such way by formula (13-8).
5. Due to the fourth event, all calculation of $\boldsymbol{M}(n)$ vector should be hold up to, e.g. (3-39), where $Q(n)$ and $R(n)$ become as arbitrary parameters.

One way on how to adapt those two arbitrary parameters is that:

6. the variance $R(n)$ will be estimated by (5-37) writing new variable $\boldsymbol{R_a}(n)$ as described in survey of Unit 5,
7. and the variance $\boldsymbol{Q}(n)$ will be adapted, writing $\boldsymbol{Q_a}(n)$, via causal rules between residuals $r(n)$ and measurements and respecting the $\boldsymbol{M}(n)$ functionality in 1., 2. and 3rd event.

According to reference [18], the diagonal $\boldsymbol{Q_a}(n)$ covariance matrix needs to be explained so that

$$\boldsymbol{Q_a}(n) = [Q(n-1) + \Delta Q(n)]\mathbf{I}\,, \tag{13-9}$$

and $\Delta Q(n)$ is the update given by adaptive process and the $\boldsymbol{Q_a}(n)$ can now be time-invariant due to realistic process noise $w(n)$.

Building up some rules to control $\Delta Q(n)$:

- The $\Delta Q(n)$ and consequent $\boldsymbol{M}(n)$ should be forced up when power of noise in residuals is **under** state-based-signal coming from the multiplication $\boldsymbol{C}\,\boldsymbol{x}(n)$, see the event 1 on page 76;
- The $\Delta Q(n)$ and consequent $\boldsymbol{M}(n)$ should be fade down when power of noise in residuals is **over** state-based-signal coming from the multiplication $\boldsymbol{C}\,\boldsymbol{x}(n)$, see the event 1 on page 76;
- The $\Delta Q(n)$ is zero and $\boldsymbol{M}(n)$ is transient unchanged when power of noise in residuals is above the state-based-signal.

These rules tend to minimised error in residuals hence an optimal state estimation in AKF. Here the problem is that a noise and $\boldsymbol{Cx}(n)$ can not be extracted from residuals. So, we need to use other evaluation criteria to determine whether the power of noise is under or over the power of clear signal. Gathering three above rules we may obtain measurable new rules:

- If residuals $r(n)$ and difference $y_v(n)$- $y_e(n)$ are correlated and associated noise $r^2(n)$ defined by (5-4) is positive and high, then state estimation accuracy is pure and the update $\Delta Q(n)$ should be very increased.
- If cross-correlation between $r(n)$ and $y_v(n)$- $y_e(n)$ is negative, then the $\Delta Q(n)$ is negative too.
- If the cross-correlation is above zero, then $\Delta Q(n)$ is zero.

The quality of $\Delta Q(n)$ in all rules is determined by a range and fuzzy sets of $r^2(n)$ as Z = *Zero*, M = *Middle* and H = *High*. As an approximation, fuzzy logic is chosen because of its simplicity.

This mechanism lends itself very well to be used in fuzzy-logic approach based on rules of the kind:

$$IF \ \{antecedent\} \ THEN \ \{consequent\}, \tag{13-10}$$

where the antecedent and consequent are of the form $\{Correlation, \ r^2(n)\} \in X_k$, $\Delta Q(n) \in L_j$, k = 1, 2, …, 6 and j = 1, 2, …, 5 respectively, where $\{Correlation, \ r^2(n)\}$ and $\Delta Q(n)$ are the input set and output variable, respectively, and X_k and Lj are their fuzzy sets. The following nine fuzzy rules of the kind (13-10) are used:

$$IF \ \{Correlation \ is \ N\} \ AND \ \{r^2(n) \ is \ Z\} \ THEN \ \{\Delta Q(n) \ is \ Z\} \tag{13-11}$$

$$IF \ \{Correlation \ is \ N\} \ AND \ \{r^2(n) \ is \ M\} \ THEN \ \{\Delta Q(n) \ is \ L-N\} \tag{13-12}$$

$$IF \ \{Correlation \ is \ N\} \ AND \ \{r^2(n) \ is \ H\} \ THEN \ \{\Delta Q(n) \ is \ N\} \tag{13-13}$$

$$IF \ \{Correlation \ is \ Z\} \ AND \ \{r^2(n) \ is \ Z\} \ THEN \ \{\Delta Q(n) \ is \ Z\} \tag{13-14}$$

$$IF \ \{Correlation \ is \ Z\} \ AND \ \{r^2(n) \ is \ M\} \ THEN \ \{\Delta Q(n) \ is \ Z\} \tag{13-15}$$

$$IF \ \{Correlation \ is \ Z\} \ AND \ \{r^2(n) \ is \ H\} \ THEN \ \{\Delta Q(n) \ is \ Z\} \tag{13-16}$$

$$IF \ \{Correlation \ is \ P\} \ AND \ \{r^2(n) \ is \ Z\} \ THEN \ \{\Delta Q(n) \ is \ Z\} \tag{13-17}$$

$$IF \ \{Correlation \ is \ P\} \ AND \ \{r^2(n) \ is \ M\} \ THEN \ \{\Delta Q(n) \ is \ L-P\} \tag{13-18}$$

$$IF \ \{Correlation \ is \ P\} \ AND \ \{r^2(n) \ is \ H\} \ THEN \ \{\Delta Q(n) \ is \ P\}, \tag{13-19}$$

where the mentioned Correlation between residual $r(n)$ and difference $y_v(n)$- $y_e(n)$ can be defined:

$$\frac{\frac{1}{25}\sum_{k=0}^{24}\left\{\left(y_v(n-k)-y_e(n-k)\right)^T r(n-k)\right\}-E\{y_v-y_e\}E\{r\}}{\sigma\{r\}\sigma\{y_v-y_e\}}. \tag{13-20}$$

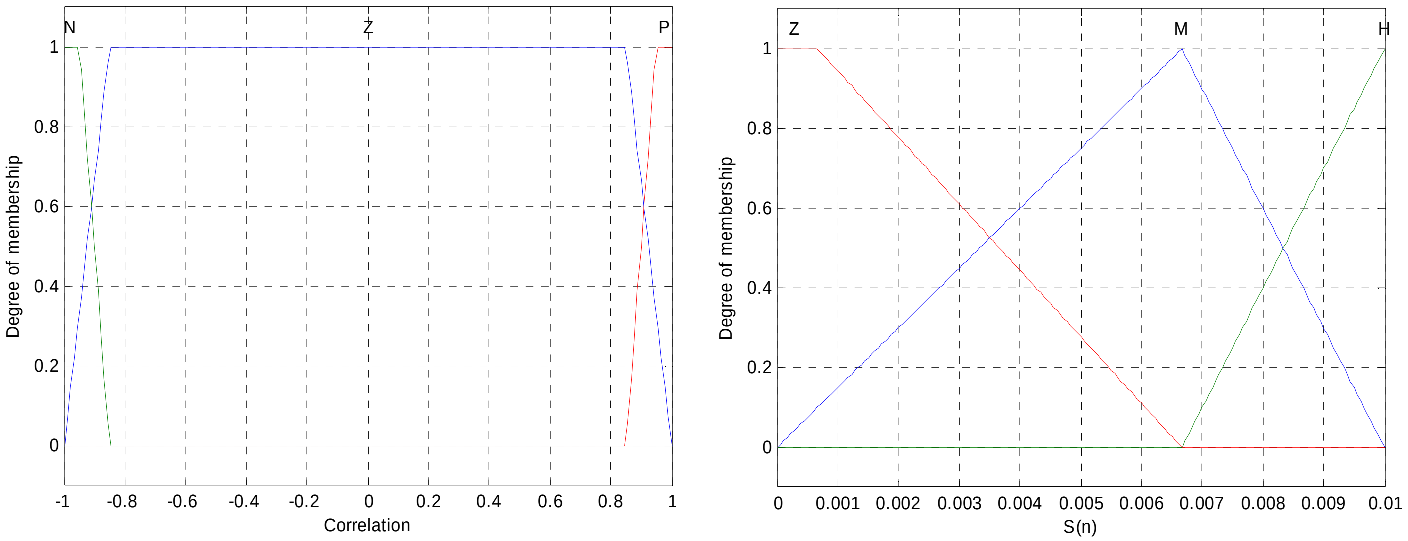
All adaptive process is computationally demanding, hence a mean E such as a smoothing filter can be computed with only two instead of 25 multiplications by using transfer function as:

$$\frac{1}{25}\frac{1-\mathrm{z}^{-25}}{1-\mathrm{z}^{-1}}. \tag{13-21}$$

The number 25 is chosen as a time of recovery when a square of transient produced in Figure 13-2 is being regenerated stable again below 5% deviation.

On the basis of the above adaptation hypothesis, the FIS can be implemented by using five fuzzy sets L for $\Delta Q(n)$: *N = Negative; L - N = Little Negative*; *Z = Zero*; *L - P = Little Positive* and *P = Positive.* For Correlation the fuzzy sets X are specified as *N = Negative, Z = Zero*; *P = Positive.*

The membership functions of these fuzzy sets are shown in Figure 13-3 to 13-4.



a) **Figure 13-3** Membership functions of **b)**

a) Correlation, and

b) $S(n) = r_1^2(n)$.

The fuzzy rules given by decision making (reasoning) of (13-11) – (13-19) can be written in table:

Correlation

	ΔQ(n)	N	Z	P
S(n)	Z	Z	Z	Z
	M	L-N	Z	L-P
	H	N	Z	P

Table 13-2 Comprehensive expression of fuzzy rules.

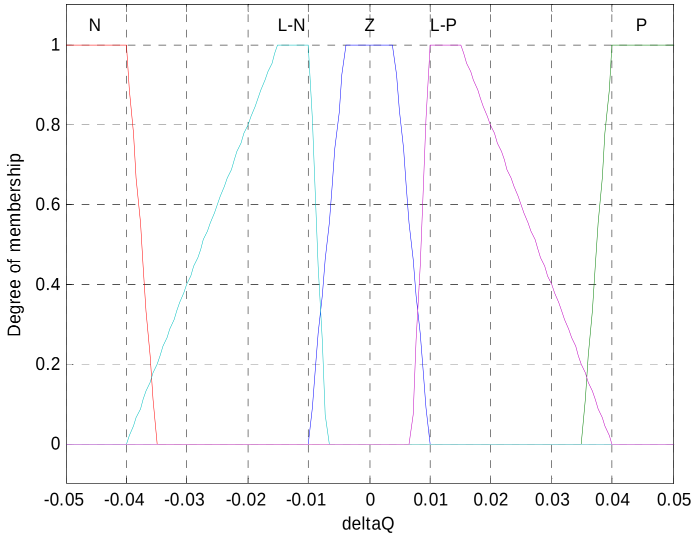


Figure 13-4 Membership functions of ΔQ(n).

Following Figure 13-5 shows surface of simulated fuzzy system referred to fuzzy rules. The profiles of fuzzy surface and membership functions are built upon trial-and-error method and can not be ruled of thumb.

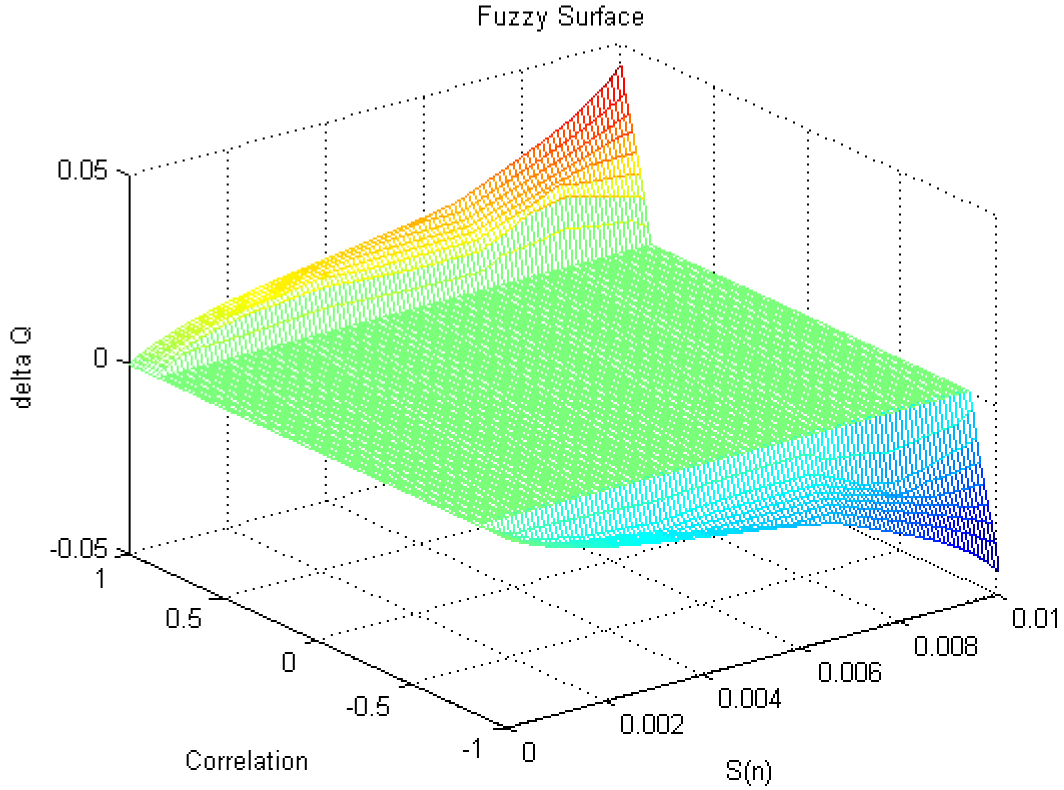


Figure 13-5 Surface produced by fuzzy logic simulation.

The inputs *Correlation* and $S(n) = r_1^{\ 2}(n)$ are within range <-1,1> and <0, 10^{-2}>, respectively. The output range of $\Delta Q(n)$ is <-$5_x 10^{-2}$, $5\,_x10^{-2}$>. The fuzzy system can be illustrated in Figure 13-6.

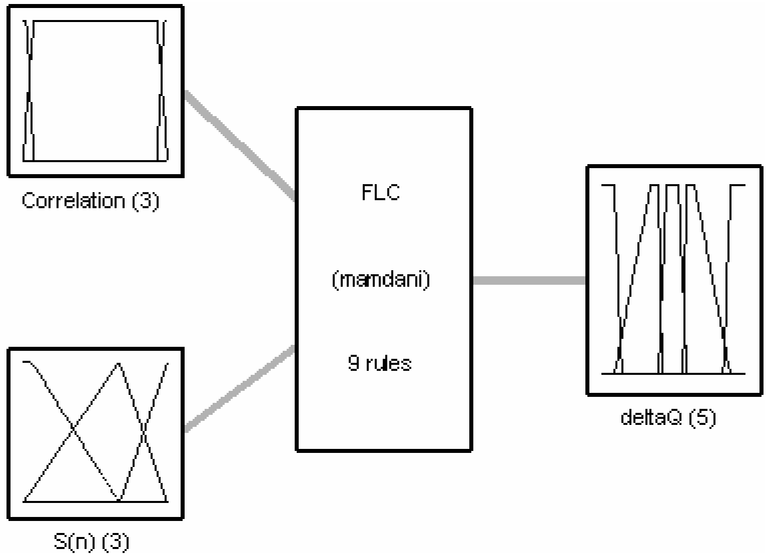


Figure 13-6 System of fuzzy logic.

These sections of adaptation problem are coupled with Unit 5 employing adaptive conventional KF, centralized and decentralized KF models. From here, the obtained $\boldsymbol{Q_a}(n)$ should be implemented into any structure of adaptive Kalman flter.

Next two sections compare decentralized AKF noticed as ADKF, and DKF inherited according to system of third and tenth order filter. This comparison focuses on state estimation accuracy of AKF.

13.1 Testing the estimator based on third order filter

This section shows results of adaptive fuzzy logic KF with one sensor. The state estimation principle refers to Figure 6-2. Generally, the ***C*** can be matrix but we use row vector as [1 0 0] that may combine three elements of $\boldsymbol{x}(n|n)$ in (5-15) to reach scalar $y_e(n)$. The state vector represents relative position of mechanical plant as a ferry or robot, and higher order derivatives such as velocity and acceleration. One can see that high order derivatives are neglected in $y_e(n)$. Thus, by a sensor measured $y_v(n)$ as position and used system model dynamics we can optimally estimate remaining two derivatives. Measured mean square error MSE in state estimates indicates accuracy presented below.

Type of KFs	**MSE measured in output filtering of KFs**	**MSE measured in state estimation of KFs**	**Q**	R_1	**Power of** w(n)	**Power of** v_1(n)
KF with constant Q, R_1	$1.343 . 10^{-3}$	$5.886 . 10^{-3}$	$3.869\ 10^{-2}$	$6.413\ 10^{-4}$	$4.139 . 10^{-2}$	$6.962 . 10^{-4}$
KF with exact Q, R_1	$1.252\ . 10^{-3}$	$5.480 . 10^{-3}$	**Q**(n)	R_1(n)	$4.139 . 10^{-2}$	$6.962 . 10^{-4}$
Adaptive fuzzy logic KF	$1.065\ . 10^{-3}$	$4.586 . 10^{-3}$	$\mathbf{Q_a}$(n)	R_{a1}(n)	$4.139 . 10^{-2}$	$6.962 . 10^{-4}$

Table 13.1-1 The third order filter test with high power of process noise w(n).

The subscript **‘1’** indicates data of first sensor. In table some results are taken into account as well as:

1. KF based on model in Table 2-1 without adaptive algorithm and with applied constant $\boldsymbol{Q}$, R_1 because process noise and sensor noise are time-invariant in following tests;
2. KF based on model in Table 2-1 without adaptive algorithm and with applied $\boldsymbol{Q}(n)$, $R_1(n)$;
3. KF based on model in Table 5-1 with adaptive algorithm employing $\boldsymbol{Q_a}(n)$, $R_{a1}(n)$.

These three alternatives will be assumed in Section 13.2.

The first measurement turns to mean square error MSE of state estimation performed by CoKF. The MSE given by (6-6) is a function of $\boldsymbol{a}(n) = \boldsymbol{x}(n) - \boldsymbol{x}(n \mid n)$, where $\boldsymbol{x}(n)$ is system state vector and $\boldsymbol{x}(n \mid n)$ is the state estimate. All three values of MSE presented by Table 13.1-1 are near to MSE computed by CoKF with exactly known $\boldsymbol{Q}(n)$ and $R_1(n)$ such as an optimal estimator. Due to unknown time behaviour of $\boldsymbol{Q}(n)$, $R_1(n)$ and $\boldsymbol{x}(n)$ this estimator can be only simulated. The MSE values are very small in relation to power of states been about 1.4.

The $y_{e1}(n)$ is expected to be filtered up to unwanted zero mean sensor noise $v_1(n)$. The Table 13.1-1 shows MSE computed with $a(n) = y_{v1}(n) - y_{e1}(n \mid n)$ argument. Those small values of MSE are about 10^{-3} toward high power of observation that is about one.

Additionally the Table 13.1-1 presents power of sensor noise 6.962_x10^{-4} and sensor noise variance is of about 6.413_x10^{-4}. Both values differ due to a little mean of sensor noise.

Following Figure 13.1-1 shows the time behaviour of $y_{e1}(n)$ filtered output :

1. denoted as $y^*_{e1}(n)$ in Kaman filter without adaptive algorithm and with applied constant $\boldsymbol{Q}$, R_1;
2. denoted as $y^{**}_{e1}(n)$ in KF without adaptive algorithm and with exactly known $\boldsymbol{Q}(n)$, $R_1(n)$;
3. denoted as $y^{***}_{e1}(n)$ in KF with adaptive algorithm employing $\boldsymbol{Q_a}(n)$, $R_{a1}(n)$.

Signals of filtered outputs are in black and should reach the target signal $\boldsymbol{Cx}(n)$ that is in blue.

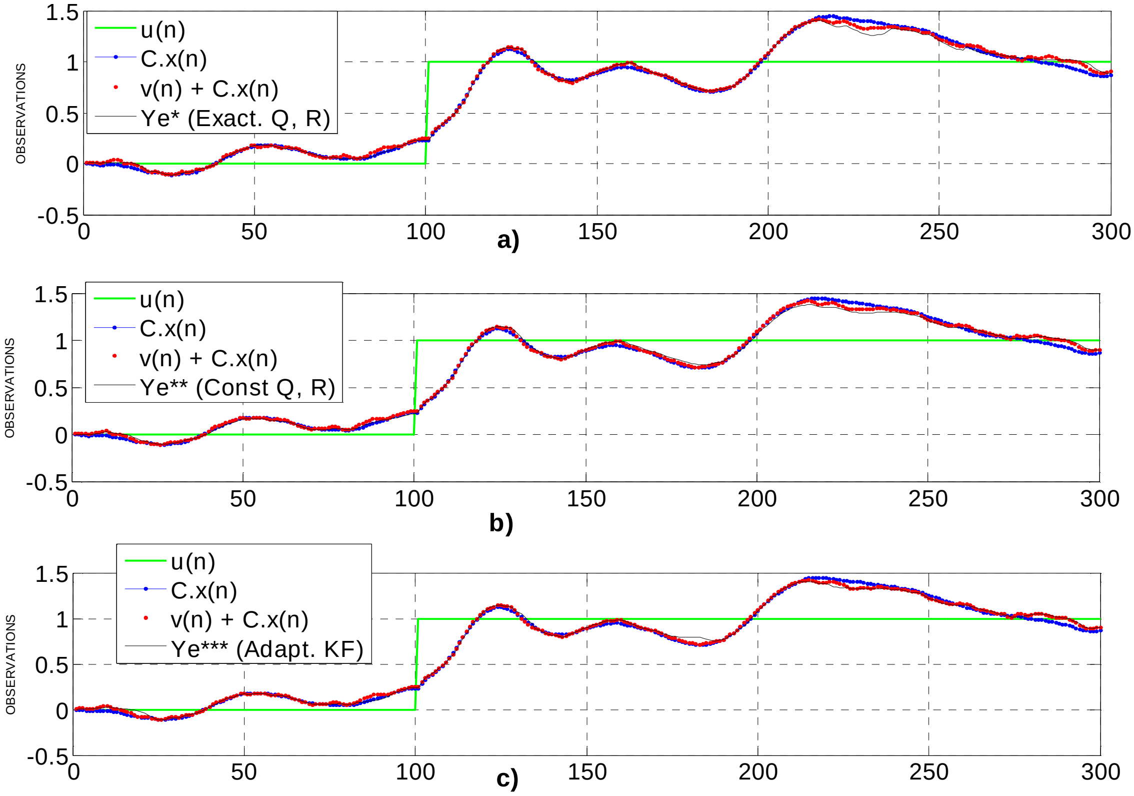


Figure 13.1-1 Test with high power of process noise. Filtered outputs:

a) $y_{e1}^{*}(n)$, *for conventional KF with exactly known Q and R_1

b) $y_{e1}^{**}(n)$, **for conventional KF with constant Q and R_1

c) $y_{e1}^{***}(n)$, ***for adaptive fuzzy logic KF.

Let us apply low power of process noise.

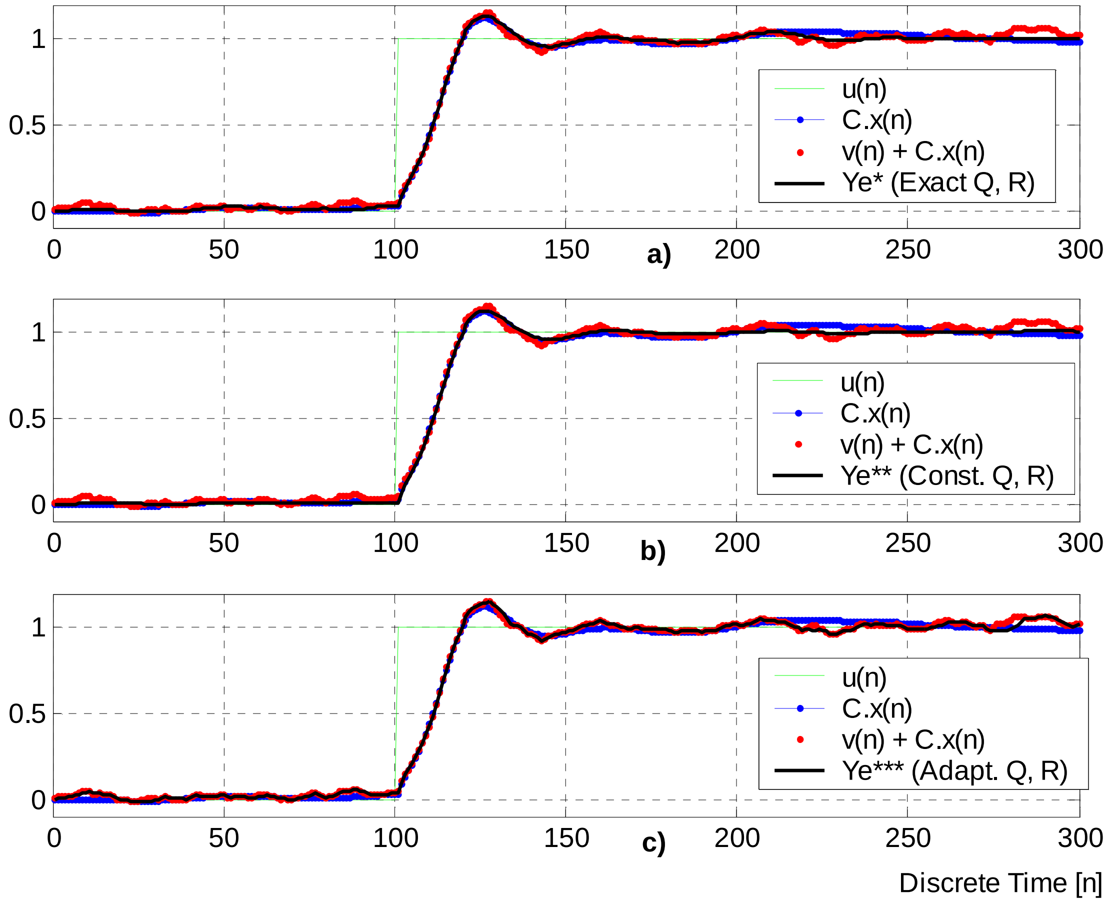


Figure 13.1-2 Test with low power of process noise. Filtered outputs:

a) $y_{e1}^{*}(n)$, *for conventional KF with exactly known Q and R_1

b) $y_{e1}^{**}(n)$, **for conventional KF with constant Q and R_1

c) $y_{e1}^{***}(n)$, ***for adaptive fuzzy logic KF.

Here, the power of process noise is 100-times lower than the used one in the previous test. After a comparison of Figures 13.1-2a and –b, the filtered outputs occur between noisy observation $y_{v1}(n)$, and clearly filtered estimation $\boldsymbol{Cx}(n)$. Figure 13.1-2c represents filtered output of adaptive fuzzy logic KF, but its output mainly follows the $y_{v1}(n)$ instead of the expected $\boldsymbol{Cx}(n)$ multiplication. This is small AKF imperfection, when the power of sensor is greater or above the power of process noise. There an anomaly occurs when $r^2(n)$ is permanently high with very high $\boldsymbol{Q_a}(n)$ to $R_{a1}(n)$ ratio. This can be overcome by any additional technique with any downtime.

Following Table 13.1-2 presents results referred to second simulation with low process noise.

Type of KFs	**MSE measured in output filtering of KFs**	**MSE measured in state estimation of KFs**	**Q**	R_1	**Power of** w(n)	**Power of** v_1(n)
KF with constant Q, R_1	3.134 10^{-4}	1.372 10^{-3}	3.869 10^{-4}	6.413 10^{-4}	4.139 10^{-4}	6.962 10^{-4}
KF with exact Q, R_1	1.877 10^{-4}	0.816 10^{-3}	**Q**(n)	R_1(n)	4.139 10^{-4}	6.962 10^{-4}
Adaptive fuzzy logic KF	6.381 10^{-4}	2.682 10^{-3}	$\mathbf{Q}_a$(n)	R_{a1}(n)	4.139 10^{-4}	6.962 10^{-4}

Table 13.1-2 The third order filter test with low power of process noise w(n).

One can see that this exercise deals with comparable power of sensor and process noise. This is causing that the MSE of adaptive KF is about 3-time higher (worse) than values obtained from conventional KF with exactly known ***Q*** and R_1.

13.2 Testing the estimator based on tenth order filter

This section investigates functionality of AKF with 10th order filter of system dynamics in similar way as previous section. The adaptive algorithm provides the highest state estimation accuracy.

Type of KFs	**MSE measured in output filtering of KFs**	**MSE measured in state estimation of KFs**	**Q**	R_1	**Power of** w(n)	**Power of** v_1(n)
KF with constant Q,R	1.378 10^{-3}	1.092 10^{1}	3.869 10^{-2}	6.413 10^{-4}	4.139 10^{-2}	6.962 10^{-4}
KF with exact Q,R	1.160 10^{-3}	0.932 10^{1}	**Q**(n)	R_1(n)	4.139 10^{-2}	6.962 10^{-4}
Adaptive KF	1.142 10^{-3}	0.923 10^{1}	$\mathbf{Q}_a$(n)	R_{a1}(n)	4.139 10^{-2}	6.962 10^{-4}

Table 13.2-1 The tenth order filter test with high power of process noise w(n).

The neglected small process noise provides 3 to 4-times worse estimation, see Table 13.2-2. Although in this test the $w(n)$ and $v_1(n)$ are very little (see Figure 13.1-2) and can not worse the state estimation at all, especially in control. Utilisation of adaptive fuzzy logic KF refers to Unit 14.

Type of KFs	**MSE measured in output filtering of KFs**	**MSE measured in state estimation of KFs**	**Q**	R_1	**Power of** w(n)	**Power of** v_1(n)
KF with constant Q,R	$2.788\ 10^{-4}$	2.566	$3.869\ 10^{-4}$	$6.413\ 10^{-4}$	$4.139\ 10^{-4}$	$6.962\ 10^{-4}$
KF with exact Q,R	$1.596\ 10^{-4}$	1.460	$\mathbf{Q}$(n)	R_1(n)	$4.139\ 10^{-4}$	$6.962\ 10^{-4}$
Adaptive fuzzy logic KF	$6.650\ 10^{-4}$	6.075	$\mathbf{Q_a}$(n)	R_{a1}(n)	$4.139\ 10^{-4}$	$6.962\ 10^{-4}$

Table 13.2-2 The tenth order filter test with low power of process noise w(n).

Very high Q to R_1 ratio may cause stability problem in Kalman filtering. Here the state estimation is mended by adaptive KF anyway in relation to unknown Q and R_1. The inappropriate Q and R_1 setting can cause practical divergence of estimation in conventional CoKF, but this has been never found at adaptive algorithm in AKF.

13.3 Conclusions

Below we apply the sixth question of section 1.1 to this conclusion to be answered:
"Test an algorithm of adaptive Kalman filter, AKF, supported by fuzzy logic to meet the Kalman filter functionality, by definition of an optimal state estimation in Kalman filtering, with a strategy of an adapted process noise covariance $\boldsymbol{Q}$ matrix and observation R_1 noise variance. Suppose both prior $\boldsymbol{Q}$ and R_1 are unknown. The reasonable effort should be made to additionally estimate the covariance matrices when the noise streams are unknown. So that, the AKF extends KF functionality to meet the practical needs when changing from traditional KF."

The Unit 13 deals with an innovation adaptive estimation, IAE, approach coupled with fuzzy logic methodology used to adjust the $\boldsymbol{Q}(n)$ diagonal matrix for state estimator. This algorithm is proposed to be implemented into models of adaptive Kalman filters presented in Unit 5. The IAE is based on innovations sequence to get an update of $\boldsymbol{Q}(n)$, and $R_1(n)$ is computed explicitly.

To exam the AKF with high power of process noise, a bit better state estimation accuracy occasionally occurs than the conventional Kalman filter provides. Considering lower power of process noise than the power of sensor noise, the $w(n)$ and $v_1(n)$ are very little and can not worse the state estimation at all, especially in control. The adaptive algorithm is considerable, otherwise an absence of that inappropriate Q and R_1 setting may cause a practical divergence of estimator, but this effect has been never found in AKF.

The algorithm of adaptive fuzzy Kalman filter has been used in single-sensor fusion. The advantage of this algorithm is utilisation in decentralized Kalman filtering, where every node processes its state estimation. Every node of decentralized structure has to share its own adapted $\boldsymbol{Q}_{ai}(n)$ covariance matrix to be transferred to its neighbourhoods (remaining nodes) where fusion must be performed according to (5-38) of Table 5-3. The adaptive algorithm implemented in centralized Kalman filter supposes the fusion of (5-28) in Table 5-2. Thus adaptive fuzzy Kalman filter can be extended to multi-sensor fusion to be dealt with next problem only in some exercises of next sections in control. In that fusion the i corresponds to i-th sensor.

14 OPTIMAL CONTROL OF LINEAR SYSTEMS

Vast use of linear system dynamics model is so in linear-quadratic regulator, LQR, that would be additionally explored with some examples. When employing any structure of Kalman filter into LQR, a usage of the estimator can be directly seen in the field of control system design. Said, this method demonstrates Linear Quadratic Gaussian LQG design. Then such devices as electrical motors, ferry and robots, etc., can be controlled and their relative position (of motor's rotor, e.g.), velocity, acceleration and high order derivatives can be estimated.

The control problem is solved in four chapters as follows.
In this unit we introduce LQR technique of linear discrete-time system and its examples. Section 14.1 examines control closed loop based on LQR additionally with nonlinear actuator. Two exercises of closed loop LQG regulator are presented in Section 14.2 with stochastic linear continuous-time system. Last Section 14.3 is the case of previous section with employed DKF and ADKF in control of multi-state-sensor system described by third order filter. Usually three components such as position, velocity and acceleration are taken into control navigations. In addition of control constraints and performance, we will focus on a process noise influence on control quality.
Finally, results of exercises mentioned above will be concluded in Section 14.4.

The concept of controlling arises when an expanded form of $u(t)$ can be changed to satisfy control conditions. Generally, a linear stochastic continuous-time system to be controlled is developed:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{F}\boldsymbol{x}(t) + \boldsymbol{\Gamma}u(t) + \boldsymbol{G}w(t), \qquad (14\text{-}1)$$

where the process noise coupling vector $\boldsymbol{G}$ is equal to $\boldsymbol{\Gamma}$ in our Simulink simulations.
The input coupling vector $\boldsymbol{B}$ has an influence of control input $u(t)$ on the state vector $\boldsymbol{x}(t)$. Two main control loops are recognised here:

- No backward feedback between $u(t)$ and $\boldsymbol{x}(t)$ is an open-loop control or feed-forward control.
- In closed-loop or feedback control the output $\boldsymbol{x}(t)$ determines the $u(t)$ in an outage loop that can not be seen in (14-1).

To mechanise closed-loop control $u[\boldsymbol{x}(t),t]$ the state $\boldsymbol{x}(t)$ must be accessible. Typically, $\boldsymbol{x}(t)$ can be directly observable only through the available observations $y_v(t)$. Designing linear control system, observations can be assumed to be given in the familiar linear expression as follows:

$$y_v(t) = \boldsymbol{C}\boldsymbol{x}(t) + v(t). \tag{14-2}$$

The concept of optimal control can be achieved when $u(t)$ is chosen to minimise performance index, or cost function, for the controlled system, [3]. The basic idea of controlled systems has been presented considering continuous-time systems in general way, [3].

One may argue that a discrete-time control system can be running on digital appliances in contrast of continues plant which model was stated in (14-1). Thus we suggest $\mathscr{Z}$ transformation. Then the control system is indeed based on flow of control as software for a processor.

The open and closed-loop control will be considered. First of all, two types of deterministic discrete-time system based on third and tenth order filter behaviour will be used in Units 14 and 14.1. To explore LQR, the third order filter can be defined by (12-1) and (12-2), where process and sensor noises have to be omitted. Tenth order filter can be given by (13-4), (13-5) and having omitted noises to get a deterministic model. There the matrices ***A***, ***B*** and ***C*** are given by (13-1) to (13-3), respectively. Those two types of system will be clearly denoted in the script. Reference input command $r_c(n)$, [5], is defined such as a step pulses :

$$r_c(n) = \begin{cases} 1 & \text{when} \quad n = 0,\ 1,\ ...,\ 100; \\ 3 & \text{when} \quad n = 101,\ 102,\ ...,\ 300 \end{cases}, \tag{14-3}$$

where time $n = 0,\ 1,\ ...,\ 300$ during all simulations. Solving the optimal control problem the input $u(n)$ will be defined as a function $u[\boldsymbol{x}(n),n]$ of $\boldsymbol{x}(t)$ and time t. Next part will present LQR technique valid generally for both types of system mentioned above.

An LQR approach to optimal control on discrete-time systems. The optimal control problem is to find the control sequence of $u(n)$ on the interval [n, N] that drives the system from a plant along a trajectory $\boldsymbol{x}(n)$ such that performance index $J(n,N)$ needs to be minimised.

To clarify the problem formulation, our control conditions might be:

a) *Minimum-Time Problems*. Let us suppose that we want to find the control $u(n)$ to drive the system from the given initial state $\boldsymbol{x}(0)$ to a desired final sate $\boldsymbol{x}$ at a minimum time.

b) *Minimum-Fuel Problems*. As mentioned in previous note, with minimum fuel at a desired time N, we would select the performance index $J = \sum_{k=0}^{N} |u(k)|$.

c) *Minimum-Energy Problems*. Just suppose that we want to find $u(n)$ to minimise the energy of the final state and all intermediate states, and also of the control used to achieve this.

We would follow the *c* criterion. Let the final time N be again fixed. In general, we could use [3], [5]:

$$J(n,N) = \frac{1}{2}\boldsymbol{x}(N)^T \boldsymbol{S}(N)\boldsymbol{x}(N) + \frac{1}{2}\sum_{k=n}^{N-1}\left(\boldsymbol{x}(k)^T \boldsymbol{V}(k)\boldsymbol{x}(k) + u(k)^T \boldsymbol{U}(k)u(k)\right), \tag{14-4}$$

where $\boldsymbol{S}(n)$, $\boldsymbol{V}(n)$ and $\boldsymbol{U}(n)$ are specified weighting matrices and N specifies final time equal to 300. The matrices $\boldsymbol{S}(n)$, $\boldsymbol{V}(n)$ are usually required to be symmetric and positive semidefinite; $\boldsymbol{V}(n)$ is symmetric and positive definite. The optimal feedback control $\boldsymbol{S}(n)$ can be defined as follows:

$$\boldsymbol{S}(k) = \boldsymbol{A}(k)^T\left[\boldsymbol{S}(k+1) - \frac{\boldsymbol{S}(k+1)\boldsymbol{B}(k)\boldsymbol{B}(k)^T \boldsymbol{S}(k+1)}{\boldsymbol{B}(k)^T \boldsymbol{S}(k+1)\boldsymbol{B}(k) + \boldsymbol{U}(k)}\right]\boldsymbol{A}(k) + \boldsymbol{V}(k). \tag{14-5}$$

In MATLAB simulations, the matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are not time functional but time-invariant. Additionally, $\boldsymbol{V}$ and scalar U are constant. The system dynamics is found out in accordance with physical problem, but all control system must be designed so that the performance index (14-4) should be chosen any to achieve the desired system response. To achieve different control objectives, different types of performance indexes in (14-4) should be selected.

As an example, we turn to system model described as the tenth order filter. Then the stationary diagonal matrix $\boldsymbol{V} = 0.05\mathbf{I}$ and $U = 1$ is found to satisfy criteria $J(n,N)$, where $\mathbf{I}$ is tenth order identity matrix. The $\boldsymbol{S}$ is computed as stationary square matrix after 10^3 backward iterations applied on (14-5). Then this steady matrix is established into (14-4). Finally, [5], the optimal control theory ensures minimised $J(n,300)$ after 301 iterations.

The matrix $\boldsymbol{V}$ and scalar U are now found to reach our control criteria:

- Required steep step response such as a short propagation time,
- Minimum overshoot under 5 % of the stable response,
- No flat but descending frequency response if possible.

Generally, pole placement is a way of selecting the LQR control closed-loop gain ***K***, but it may not always be satisfactory because good set of pole locations is not always obvious. One way to select the gain (or optimal feedback gain) that alleviates mentioned difficulties is to minimise (14-4) quadratic performance criterion (index or cost function) during the time interval of interest [n, 300]. By a given system model equation we obtain the gain for linear state feedback control law, [5] and [3] :

$$u(n) = -\boldsymbol{K}(n)\boldsymbol{x}(n) \tag{14-6}$$

that minimises the quadratic performance criterion (14-4), where ***K***(*n*) is gain sequence:

$$\boldsymbol{K}(k) = \left[\boldsymbol{B}(k)^T \boldsymbol{S}(k+1)\boldsymbol{B}(k) + \boldsymbol{U}(k)\right]^{-1}\left[\boldsymbol{B}(k)^T \boldsymbol{S}(k+1)\boldsymbol{A}(k)\right],\ k < N\,. \tag{14-7}$$

Generally, this ***K***(*k*) is in iterations based gain which final ***K***(*n*) can be determined in various ways of:

- Pole placement of the closed-loop system,
- Optimisation of a quadratic performance criterion (14-4).

If system model is controllable, the closed-loop poles can be placed wherever in complex unit plane.

The optimal feedback control ***S***(*k*), and subsequently LQR closed-loop control gain ***K***(*k*) can be computed as a sequence shown in Figure 14-1. There the ***K***(*k*) sequence is provided from backward iterations, but only the steady solution ***K*** is needed. We suppose computed stationary squared ***S*** matrix beforehand. Then the ***K*** is computed by using formula (14-7). The LQR closed-loop control gain ***K*** is constant vector as well as ***A*** matrix and ***B*** vector do, and *U* is scalar.

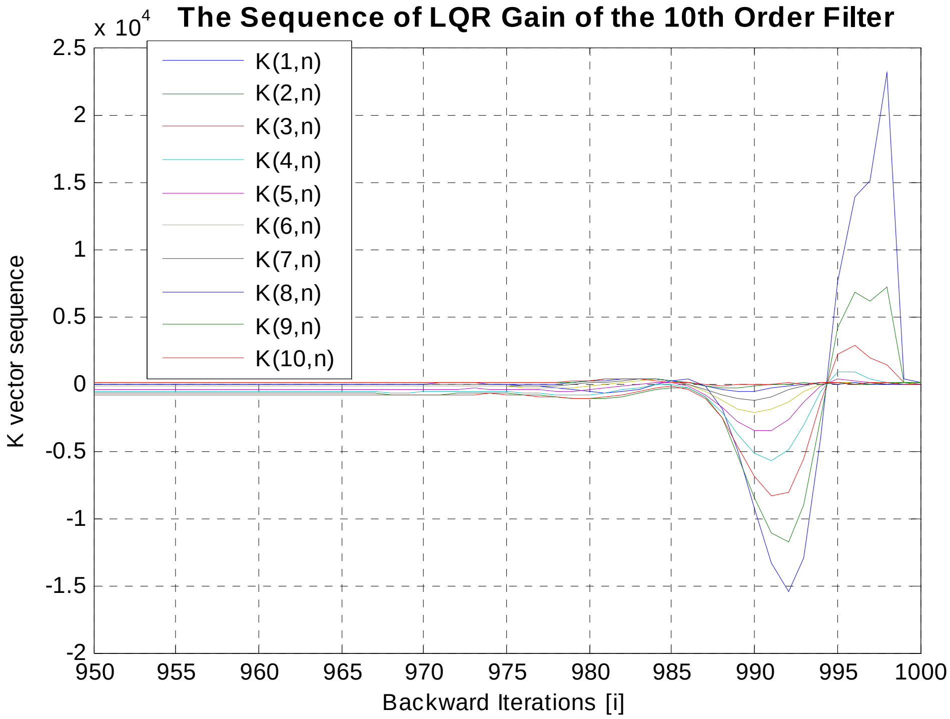


Figure 14-1 Sequence of LQR closed-loop control gain for tenth order filter.

In case of tenth order filter, the computed values of gain $\boldsymbol{K}$ are shown:

$$\left[-155.9795\ -775.2659\ -821.6757\ -649.0470\ -431.8695\ -244.1280\ -108.8060\ \ -24.5893\ \ \ 19.9992\ \ \ 37.8702\right] \quad (14\text{-}8)$$

After the gain has been applied to closed-loop discrete-time control system, then the model of system dynamics can be rewritten to, see [5] and [2] :

$$\boldsymbol{x}(n+1)=(\boldsymbol{A}-\boldsymbol{BK})\boldsymbol{x}(n). \qquad (14\text{-}9)$$

As the control closed-loop is designed in MATLAB, new state transition matrix can be found out of braces in (14-9) by writing new variable $\boldsymbol{A_c}$ been explored below,

$$\boldsymbol{Ac}=\boldsymbol{A}-\boldsymbol{BK}\,. \qquad (14\text{-}10)$$

Roughly spoken, we can say that the proposed closed-loop feedback changes transfer function of system. This fact is shown in Figure 14-2 by depict frequency and phase response of $\boldsymbol{x}(1,n)$.

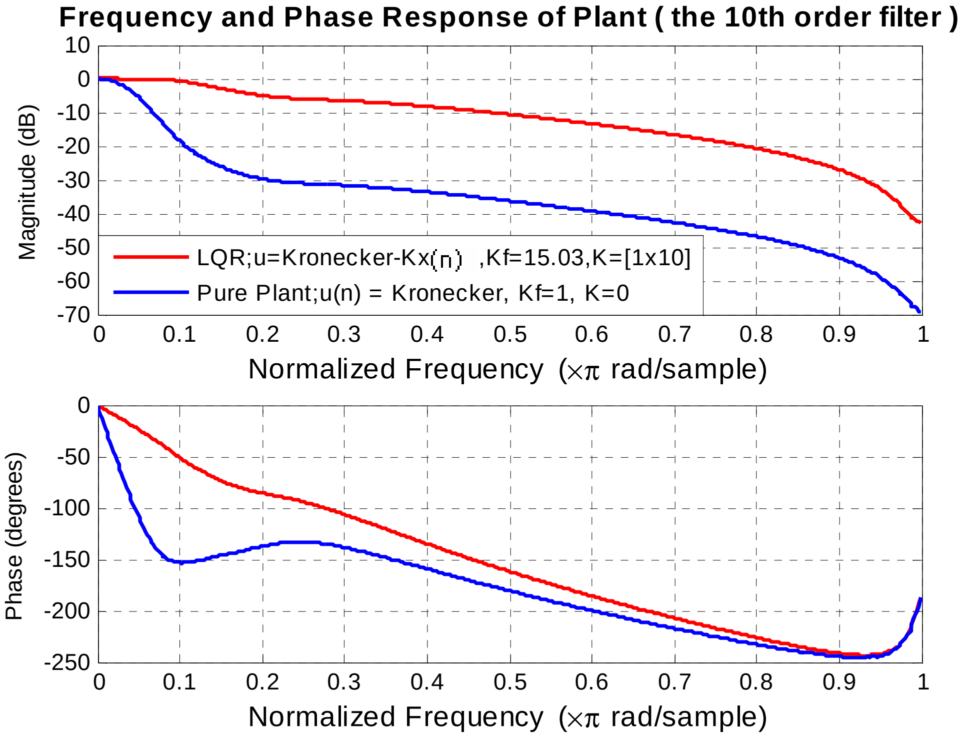


Figure 14-2 Frequency and phase response of LQR closed-loop system with tenth order filter.

The blue curves represent behaviour of system (13-4), said without LQR control closed-loop. There the phase response is nonlinear and magnitude is dominated in low frequencies mainly. Magnitude and phase response of the controlled system are more flat, see the red curves.

Full state-feedback control can be defined by

$$u(n) = K_f r_c(n) - \boldsymbol{Kx}(n), \tag{14-11}$$

Where $r_c(n)$ is reference input command and K_f is the feed-forward gain. The gain $\boldsymbol{K}$ (14-8) has been selected by LQR technique, with the respect to algorithm presented in [5]. Generally, K_f must be found to satisfy the stationary state $\boldsymbol{x}(\infty)$ if $r_c(\infty)$ is also stationary. All examples the time n is limited up to 300 and the scalar K_f =15.03.

The LQR closed-loop control system in discrete-time domain is presented by schema in Figure 14-3. In MATLAB simulations the following expression is taken a form of $u(n)$-$\boldsymbol{Kx}(n+1-1)$ instead of $u(n)$-$\boldsymbol{Kx}(n)$, where $\boldsymbol{x}(n+1)$ is the time update of system and sign '-1' represents the unit delay. Practically, if no using this expression, an algebraic loop happens.

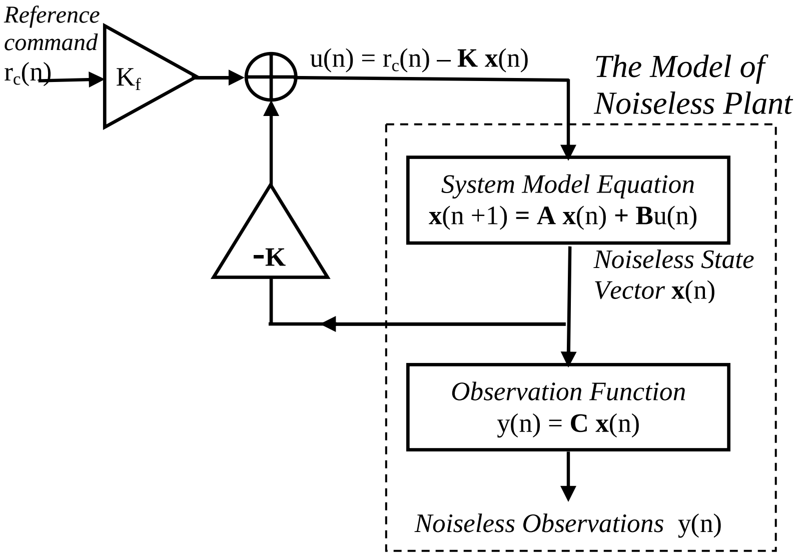

Reference command
$r_c(n)$
K_f
$u(n) = r_c(n) - \mathbf{K}\,\mathbf{x}(n)$
The Model of Noiseless Plant
System Model Equation
$\mathbf{x}(n+1) = \mathbf{A}\,\mathbf{x}(n) + \mathbf{B}u(n)$
-K
Noiseless State Vector $\mathbf{x}(n)$
Observation Function
$y(n) = \mathbf{C}\,\mathbf{x}(n)$
Noiseless Observations y(n)

Figure 14-3 Schema of LQR closed-loop control system.

The Figure 14-3 describes only a control model, because the states $\boldsymbol{x}(n)$ are not obtained directly from a plant. The object is to demonstrate LQR theory and (14-4) formula that operates in simulation. The term "linear-quadratic" refers to the linear system dynamics and the quadratic cost function. Note that both the plant and the cost-weighting matrix $\boldsymbol{V}$ and U (the state and control penalty matrices) can be time-varying, see [5]. The initial plant state is given as $\boldsymbol{x}(0) = \boldsymbol{0}$. We assume $U > 0$ for all iterations n. The aim of LQR is to find control sequence $u(n)$, $u(n+1)$, … , $u(N-1)$ that minimises $J(n,N)$. The $\boldsymbol{x}(N)$ is so-called boundary state vector. The matrix $\boldsymbol{V}$ is selected to give an appropriate weighting of the deviation of the state $\boldsymbol{x}(N)$ from the origin, and U is chosen to limit the control effort $u^T u$. The quadratic performance criterion (14-4) is a compromise between its minimum and control criteria.

Only defaulted ***V*** to *U* ratio is important and if another ratio is larger than the defaulted one, then deviations of $\mathbf{x}(n)$ from $\mathbf{x}(N)$ will be penalised heavily relative to deviations of $u(n)$ from $u(N)$. On the other hand, if the control energy been small is important, then we should select a lover ratio than the defaulted one, and the state will not quickly converge to $\mathbf{x}(N)$. In our case, the ***V*** to *U* ratio matters for finding the LQR closed-loop (state-feedback) gain ***K***. After many performed iterations of ***S*** and consequent ***K*** computation satisfied the control criteria. The blue or red curve depicts $y_{v1}(n) \equiv \mathbf{x}(1,n)$ response of plant with or without LQR, respectively in Figure 14-4. The curve of $r_c(n)$ reference input command is shown in green. This plant's time behaviour $\mathbf{x}(1,n)$ is caused by $u(n)$ shown in Figure 14-5. Expression $y_{v1}(n) \equiv \mathbf{x}(1,n)$ is valid in Unit 14 to 14.1.

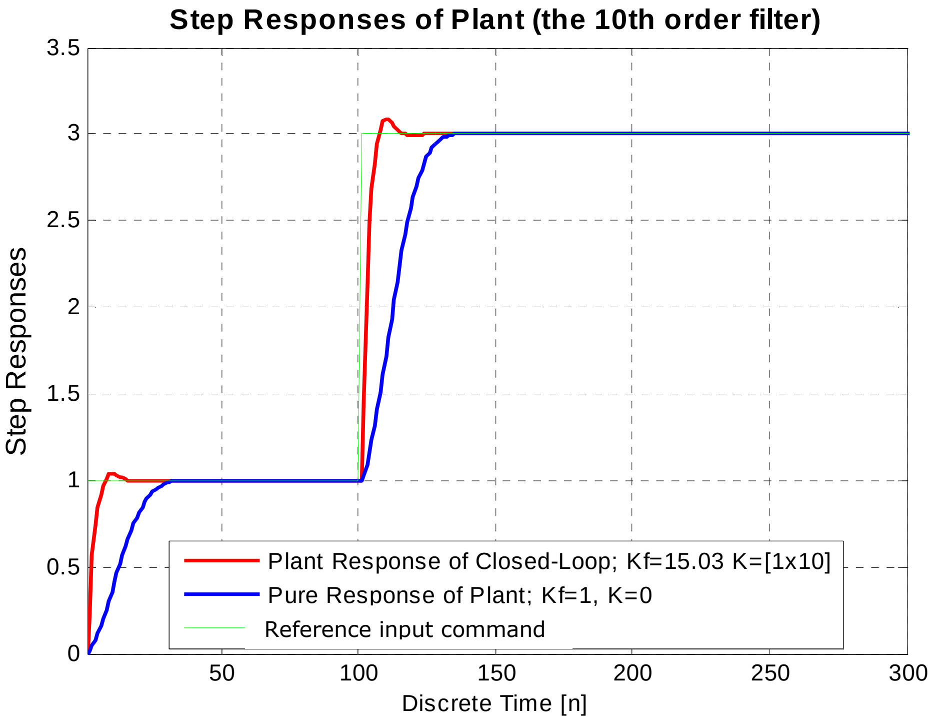


Figure 14-4 Step responses of system with/without LQR, $r_c(n)$ in green.

Both Figures 14-4 and 14-5 describe LQR control system performance.

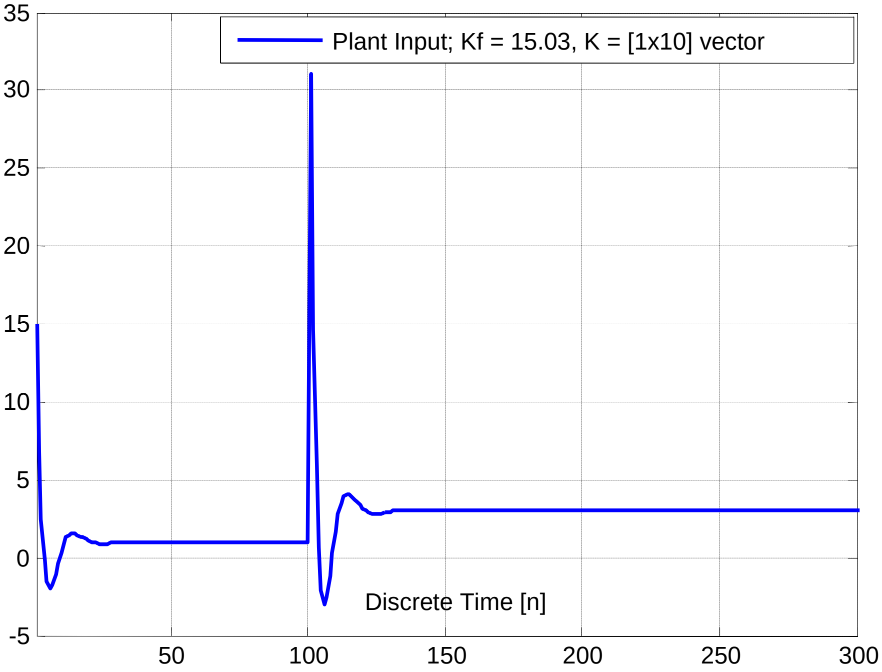


Figure 14-5 Input of plant in closed-LQR-loop (the 10th order filter).

When using third order filter as system model dynamics, stationary diagonal matrix should be selected ***V*** been equal to 0.098**I** and $U = 1$ to fully satisfy control criteria mentioned on page 90. The system dynamics and appropriate noises are clarified especially for LQR control system on page 89.

The sequence of LQR closed-loop control gain ***K*** is obtained by using (14-5) and (14-7).

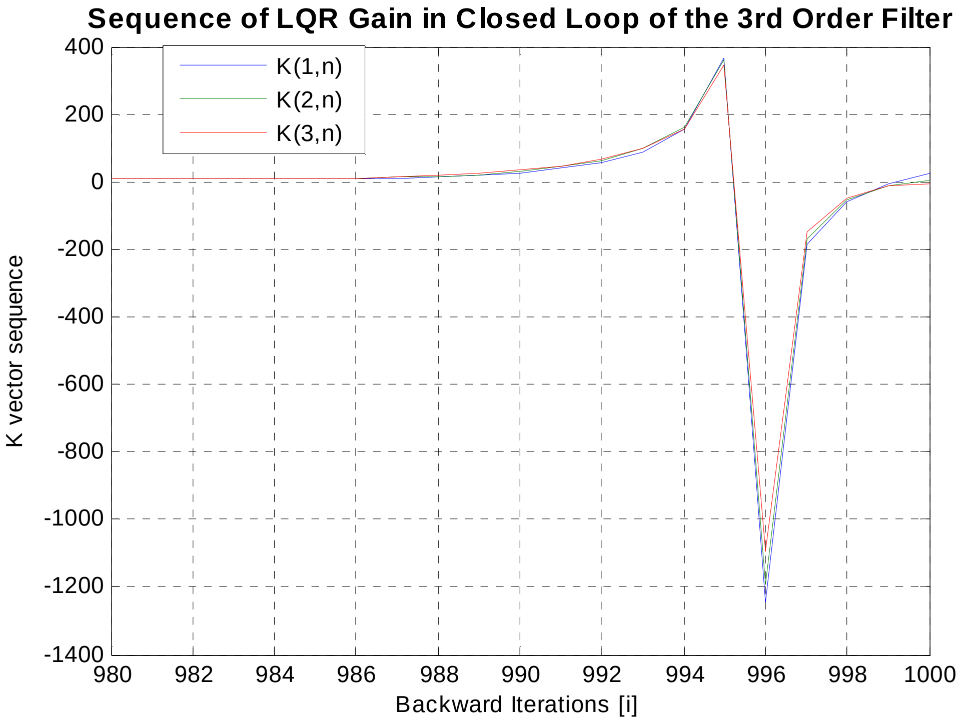


Figure 14-6 Sequence of LQR closed-loop control gain for third order filter.

The gain ***K*** is found as

$$[3.8 \quad 3.789 \quad 3.74]. \tag{14-12}$$

Feed-forward gain K_f = 10.206. Following the schema in Figure 14-3, noiseless observations response of plant is depict in Figure 14-7. Observations signal $y_{v1}(n)$ of system with LQR satisfies the control criteria in which the overshoot is less than 5% and the form of steep response has been achieved. The input control signal of plant is depict in Figure 14-8.

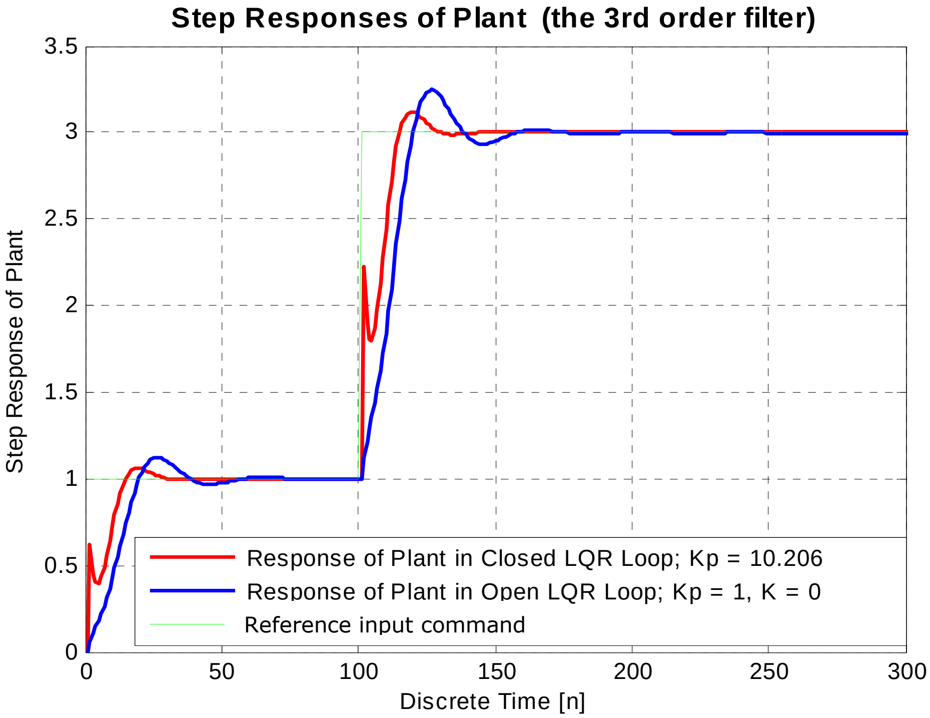


Figure 14-7 Step responses of system with/without LQR, $r_c(n)$ in green.

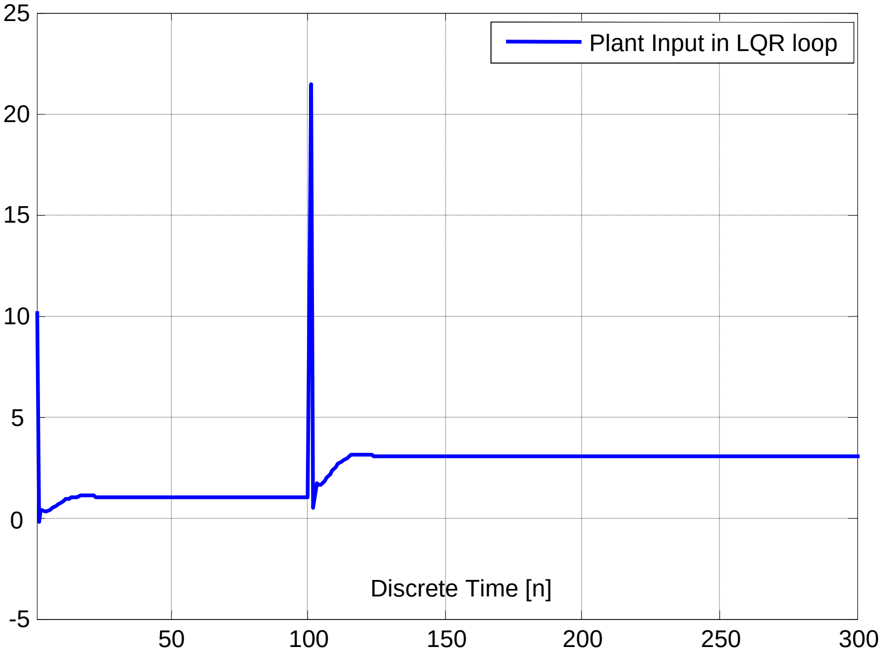


Figure 14-8 Input of plant in closed LQR loop (the 3rd order filter).

The third condition of control objectives is satisfying quite badly as shown in Figure 14-9.

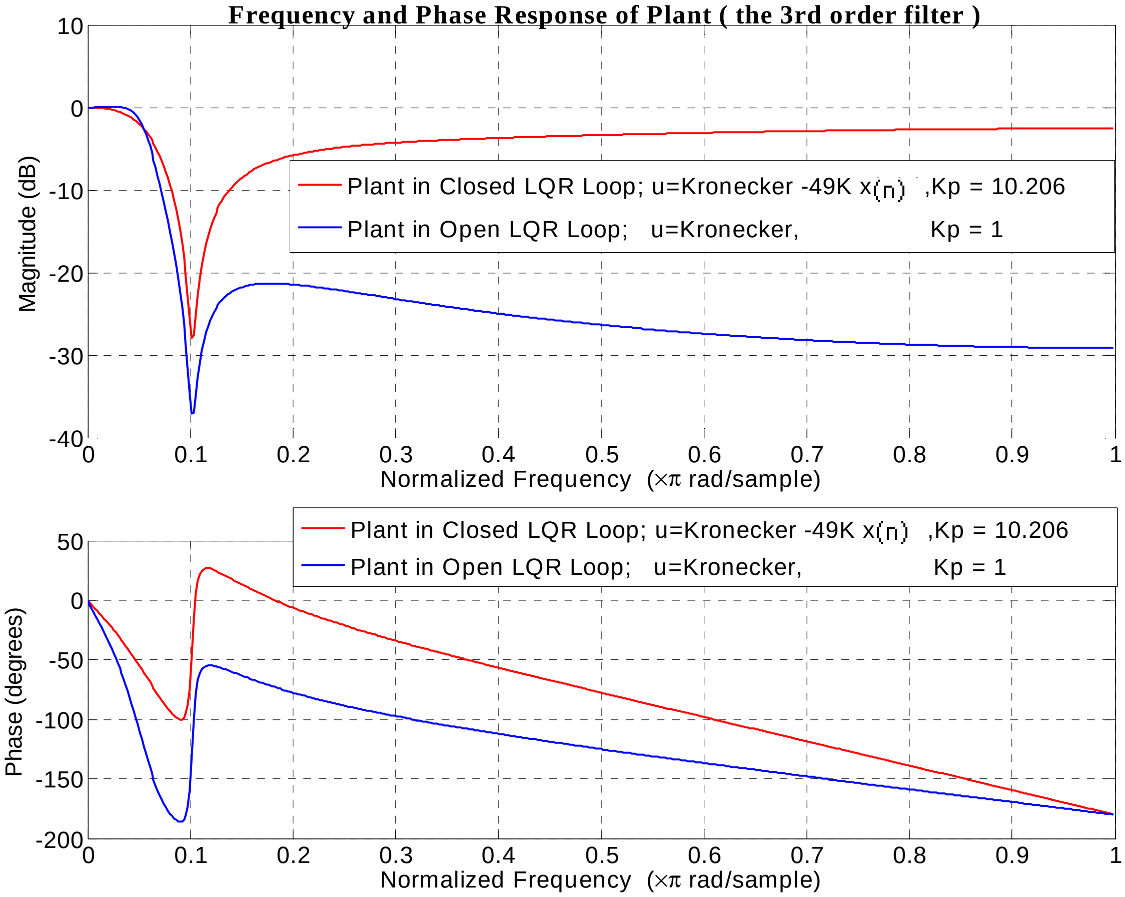


Figure 14-9 Frequency and phase response of LQR closed-loop system with third order filter.

The LQR gain (14-11) must be additionally multiplied by 49 to make the control conditions completely satisfied. To fulfil the first condition, high frequency band is dominated and flat excepting zero in transfer function at normalised frequency 0.1. In contrast, the blue curve hyper-belongs to pure simulation of plan/system without any LQR.

14.1 Examples in Linear Quadratic Regulator using a nonlinear actuator

An exercise presents the focus on how to overcome undesirable negative range coming to an actuator in modified LQR. The actuator provides positive output as $u(n)$ assumed to be e.g. a heat in air conditioning system without cooling (negative range – outcoming energy). As one can see in Figure 14-5, the control input $u(n)$ of plant comes into negative range. If this range is saturated to zero by an actuator, the gain $\boldsymbol{K}$ (14-8) based on LQR close-loop produces high overshoots. To ensure LQR operating in linear (positive) range of $u(t)$, the control system must behave in linear filtering. Hence we like to find a new LQR closed-loop gain to solve mentioned problem with reference input command $r_c(n)$ given by (14-3). Of course this example deals with new time behaviour of $r_c(n)$ and LQR gain:

$$[1836 \quad 298.4 \quad -440.5 \quad -708.1 \quad -720.4 \quad -612.9 - 465.9 \quad -322.9 \quad -203.4 \quad -113.3], \quad (14\text{-}13)$$

where feed-forward gain $K_f = 7.21$ approximately. The LQR closed-loop gain $\boldsymbol{K}$ has been found due to selected stationary diagonal matrix $\boldsymbol{V}$ is equal to $8.81_x 10^{-3}$ $\mathbf{I}$ and $U = 1$ to satisfy control objectives :

1. Steep response such as a short propagation time,
2. Minimum overshoot been less than 5 % of the stable value,
3. Least flat or descending frequency response,
4. Undesirable negative excitation in $u(n)$ has to be overcome by (14-13) in modified LQR control system.

The new concept introduced to (14-13) shows the $y_{v1}(n)$ that is a response in modified LQR with nonlinear actuator, see red curve in Figure 14.1-1. The typical response shown in dashed curve is provided with fully linear LQR control system, where the $r_c(n)$ has not limited range with appropriate $\boldsymbol{K}$, see the expression (14-8) and Figure 14-4. These two responses differ. The controlled system with nonlinear actuator gives a bit slower response than the controlled system with full actuator's range, although the control criteria are satisfied. The input control signal $u(n)$ is presented in Figure 14.1-2. There one can see that the blue curve never crosses the unwanted negative range. Thus the linear filtering is ensured in this case on actuator's nonlinearity, and input control signal does not reach negative values.

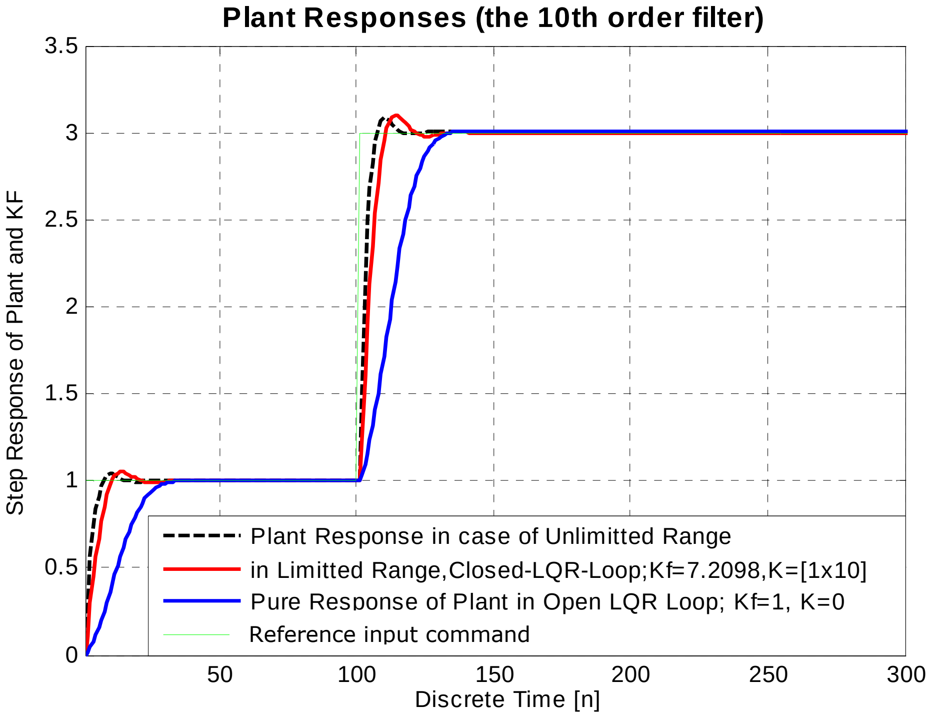


Figure 14.1-1 Step responses of system with/without LQR, $r_c(n)$ and new response.

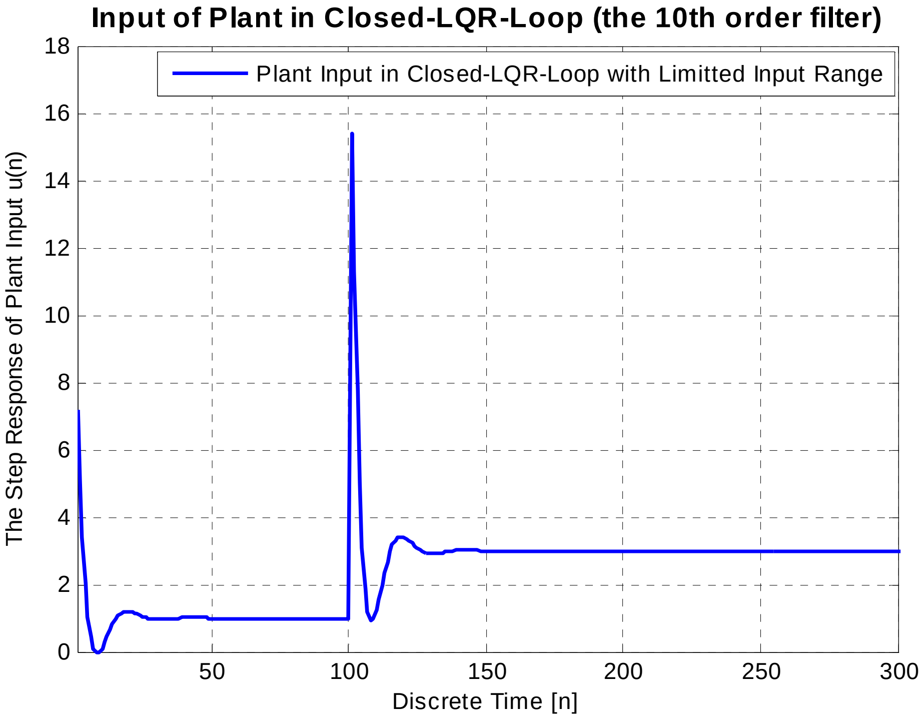


Figure 14.1-2 Input of plant in closed LQR loop (new gain, the 10th order filter).

A fuzzy gain scheduler might be needed due to a nonlinear actuator, if $r_c(n)$ is not restricted to (14-3).

14.2 Examples in Linear Quadratic Gaussian design

This section focuses on some exercises of proposed regulator in linear-quadratic Gaussian, LQG, design. Two stochastic discrete-time systems will be implemented in LQG and tested separately. The first system follows the behaviour of tenth order filter to be called a system of tenth order, which:

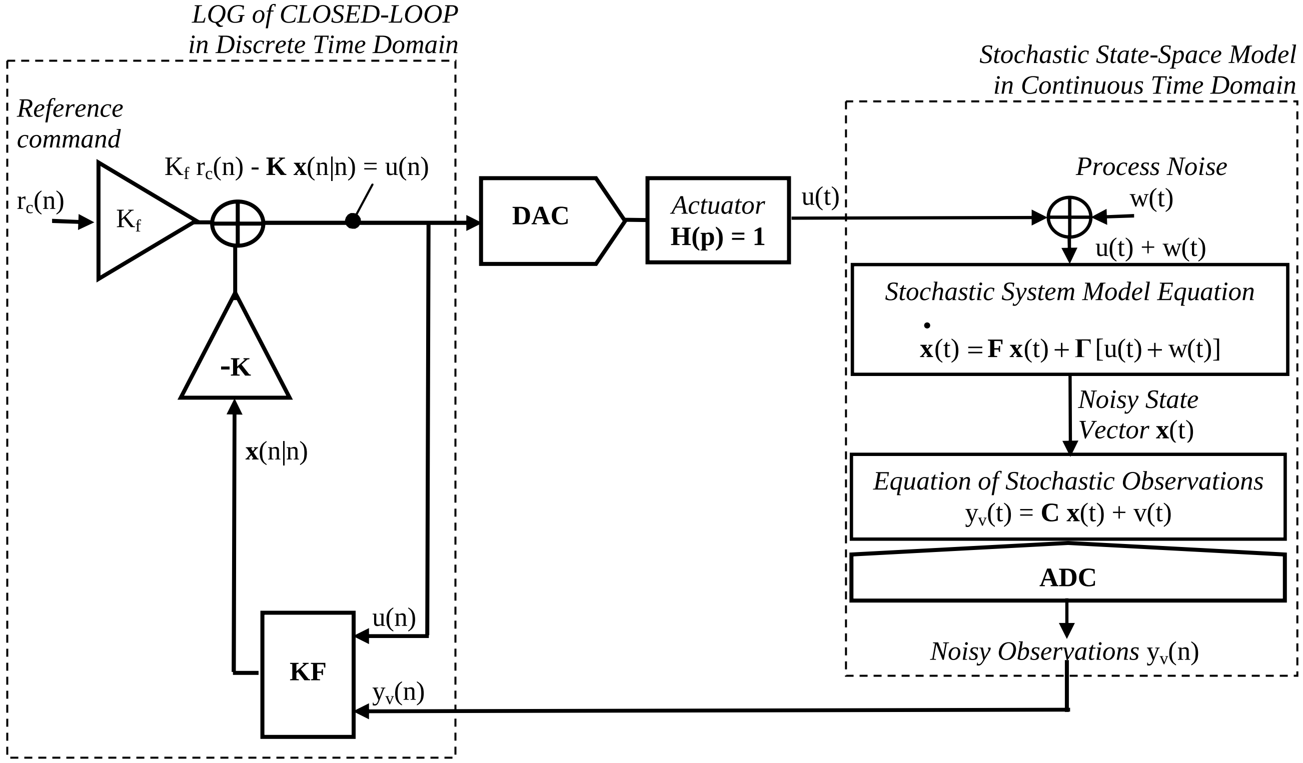
- system model equation is given by (13-4) and
- observations (sensor) model equation is given by (13-5), where considered ***A***, ***B*** and ***C*** matrices are given by (13-1) to (13-3), respectively.

The second system of third order can be described by:

- system model equation is given by (12-1), and
- observation model equation is given by (12-2).

Observation and sensor noises are now fully employed. The Unit 6 deals with those noises models to be used here. The process noise and sensor noises are uncorrelated and their variances are specified in Unit 6. The $w(n)$ noise is only one stream used the same in all simulations. Reference input command $r_c(n)$ is given by (14-3). We have to note that the expression $y_{v1}(n) = \boldsymbol{C}_1 \boldsymbol{x}(n) + v_1(n)$ due to stochastic systems mentioned above been valid up to Section 14.3. This expression provides a model of first observation of a plant, where $y_{v1}(n)$ and $\boldsymbol{x}(n)$ will be investigated.

In complicate modern systems may be no priori guidelines for selecting the compensator structure. In this case, a combination of linear-quadratic state-feedback regulator and state estimator proves very usefulness. This combination is known as linear-quadratic Gaussian design, and its structure is shown in Figure 14.2-1. There, the LQR problem and Kalman filter can be coupled together to design LQG. In mathematics, this procedure is called linear-quadratic Gaussian problem. An important advantage of LQG design is that the compensator structure is given by procedure that needs not be known beforehand. This makes LQG design useful in the control of e.g., aircraft engines, helicopters, space structures and robotics, where an appropriate compensator structure may be not known. By referring to Figure 14.2-1, the LQR's state-feedback gain ***K*** was selected by LQR technique in Unit 14. The Kalman filter and its innovation KF gain ***M***(*n*) was found in either Unit 2 – 5 or 6. Both gains were designed separately to yield closed-loop plant behaviour and observer behaviour. This is the separation principle which is at the heart of modern controls design. Two important notifications of the separation principle are made when close-loop stability is guaranteed and good software is available. The first main idea of structure in Figure 14.2-1 is to perform plant control, second that the KF should estimate plant states optimally. The LQG described briefly above can be running on a digital *signal* processor, consequently that is the reason of shown LQG part in discrete-time domain and plant naturally in continuous-time domain.



LQG of CLOSED-LOOP
in Discrete Time Domain
Reference
command
$K_f\ r_c(n) - \mathbf{K}\ \mathbf{x}(n|n) = u(n)$
$r_c(n)$
K_f
-K
$\mathbf{x}(n|n)$
u(n)
KF
$y_v(n)$
DAC
Actuator
H(p) = 1
u(t)
Stochastic State-Space Model
in Continuous Time Domain
Process Noise
w(t)
u(t) + w(t)
Stochastic System Model Equation
$\dot{\mathbf{x}}(t) = \mathbf{F}\ \mathbf{x}(t) + \mathbf{\Gamma}\,[u(t) + w(t)]$
Noisy State
Vector $\mathbf{x}(t)$
Equation of Stochastic Observations
$y_v(t) = \mathbf{C}\ \mathbf{x}(t) + v(t)$
ADC
Noisy Observations $y_v(n)$

Figure 14.2-1 Closed-loop schema in LQG design.

To evaluate the states in plant from the above structure of LQG, we need to specify the mean square error – MSE and root mean square error – RMSE as follows:

$$MSE = \frac{1}{300}\int_{t=0}^{300}\left(\boldsymbol{x}(t)^{Closed\text{-}Loop} - \tilde{\boldsymbol{x}}(t)^{Closed\text{-}Loop}\right)^T\left(\boldsymbol{x}(t)^{Closed\text{-}Loop} - \tilde{\boldsymbol{x}}(t)^{Closed\text{-}Loop}\right)dt \Big/ \int x^T x dt\,, \qquad (14\text{-}14)$$

$$RMSE = \sqrt{MSE}\,, \qquad (14\text{-}15)$$

where the denominator simplifies the expression for power of states $\boldsymbol{x}(t)^{Closed\text{-}Loop}$. This states vector is loaded from plant in LQG closed control loop with KF and exactly known ***Q*** and *R*. To employ AKF in LQG, the tilde is marked in (14-14). Additionally we focus on a performance of open-loop so-called feed-forward control structure shown in Figure 14.2-2.

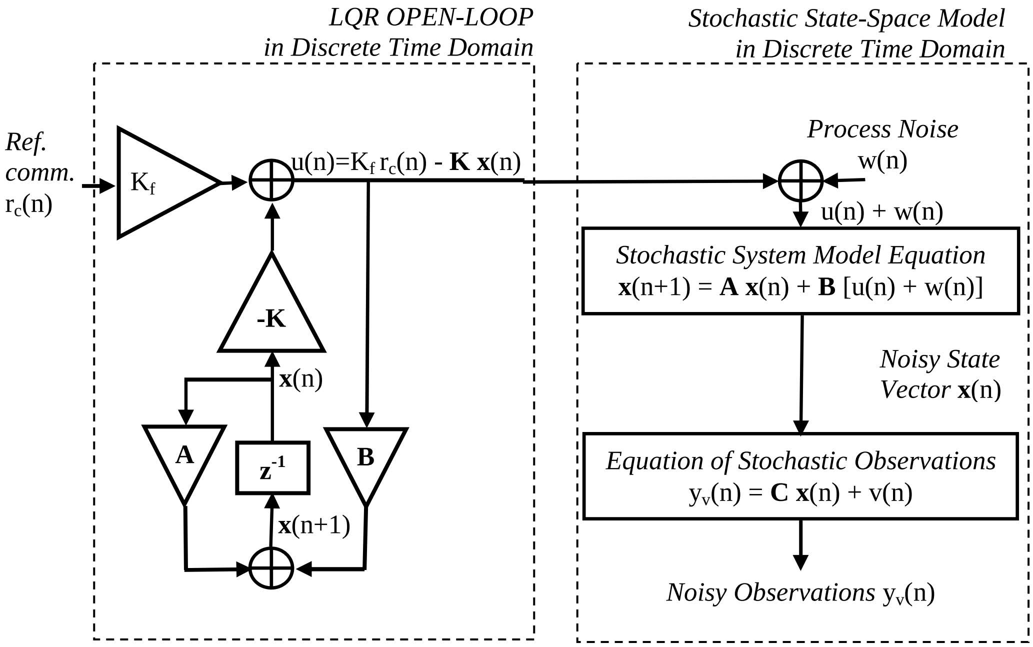


Figure 14.2-2 Open-loop schema in LQR design.

The gin ***K*** was found as:

1. system of tenth order, where corresponded control gain is given by (14-8);
2. system of third order, where corresponded vector (14-11) times 49.

Further statement considers Figures 14.2-3 and 14.2-4. Every simulation provides some results of:

a) LQR that controls the noiseless plant to be deterministic, with zero process and sensor noise; see green curve in figures; the plant provides $\boldsymbol{x}(t)^{NoiseLess\ \ Plant}$ and $y_{v1}(t)^{NoiseLess\ \ Plant}$,

b) stochastic plant controlled by open-loop LQR; see red curve in figures; here the plant provides $\boldsymbol{x}(t)^{Open\text{-}Loop}$ and $y_{v1}(t)^{Open\text{-}Loop}$,

c) stochastic plant controlled by closed-loop LQG regulator, where KF with exactly known $\boldsymbol{Q}(n)$ and $R(n)$; see blue curve in figures,

d) stochastic plant controlled by closed-loop LQG regulator, where adaptive fuzzy Kalman filter (see Unit 13) is utilised; see black curve in figures, here the plant provides $\boldsymbol{x}(t)^{Closed\text{-}Loop}$ and $y_{v1}(t)^{Closed\text{-}Loop}$.

Practically the benefit on LQG design is coupled with last *d* option. A result referred to option *a* is the target being never achieved.

Option of *b* refers to pure control with open-loop LQR. This kind of regulator is established because of simulation and regulators comparison. There a stochastic system model with $w(n)$ and $v_i(n)$ noises is supposed, where $i = 1, 2, \ldots, 10$ (or 3).

Last two options *c* and *d* refer to the stochastic systems have been discussed, but a state-closed-loop LQG design is applied in control. System states should be numerically near.. Additionally, it is expected that an uncertainty extracted from those states should be a little toward an uncertainty produced by plants state in option *b*. This fact should happen due to numerically very strong LQR gain ***K*** that meaning high values of that gain. Intuitively, this strong gain causes high feed-forward K_f gain. This kind of attenuation can be measured only in simulations. Of course main idea of LQR and LQG/LTR design is nothing but optimal control. The gain ***K*** attenuates process noise in states and this capability is a by-product and can be obtained from measured state in plant and observations respectively as follows:

$$NNRx = \frac{\sum_{i=0}^{300}\left(x(n)^{NoiseLess\ \ Plant} - x(n)^{Closed\text{-}Loop}\right)^T\left(x(n)^{NoiseLess\ \ Plant} - x(n)^{Closed\text{-}Loop}\right)}{\sum_{i=0}^{300}\left(x(n)^{NoiseLess\ \ Plant} - x(n)^{Open\text{-}Loop}\right)^T\left(x(n)^{NoiseLess\ \ Plant} - x(n)^{Open\text{-}Loop}\right)} \qquad (14\text{-}16)$$

$$NNRy = \frac{\sum_{i=0}^{300}\left(y_{v1}(n)^{NoiseLess\ \ Plant} - y_{v1}(n)^{Closed\text{-}Loop}\right)^2}{\sum_{i=0}^{300}\left(y_{v1}(n)^{NoiseLess\ \ Plant} - y_{v1}(n)^{Open\text{-}Loop}\right)^2}. \qquad (14\text{-}17)$$

An assessment of a performance of LQG design based regulator and tenth order filter.

This part deals with performance of LQG design based regulator, Figure 14.2-1. Some results of that regulator will be compared to some results of open-loop regulator, Figure 14.2-2. In this section, LQG regulator has two modifications, such as to be with adaptive fuzzy logic Kalman filter (as AKF, see Table 5-1) and conventional Kalman filter CoKF which bases on exactly known covariance of process and sensor noise. Both estimators provide very similar observations $y_{v1}(n)$. This first observation describes relative position of ferry, robot or motor's rotor, etc. The second modification can be of course only simulated and is useful such as reference regulator solving linear-quadratic Gaussian and loop-transfer recovery (LQG/LTR) problem, [5]. To be honest, the $r_c(n)$ is Hevisideon pulse Һ(n-100). The green curves in below figures represent observations of open-loop regulator, see Figure 14.2-2, but without any noise. The same is illustrated in red with process and sensor noises in plant.

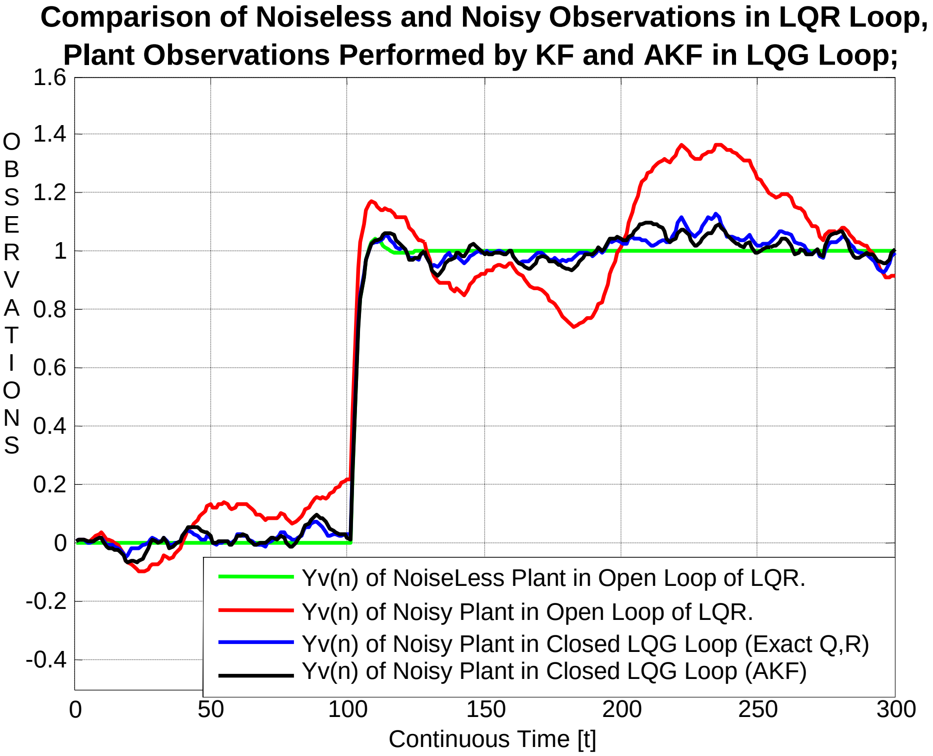


Figure 14.2-3 Observations obtained from different regulators with high power of process noise in plant.

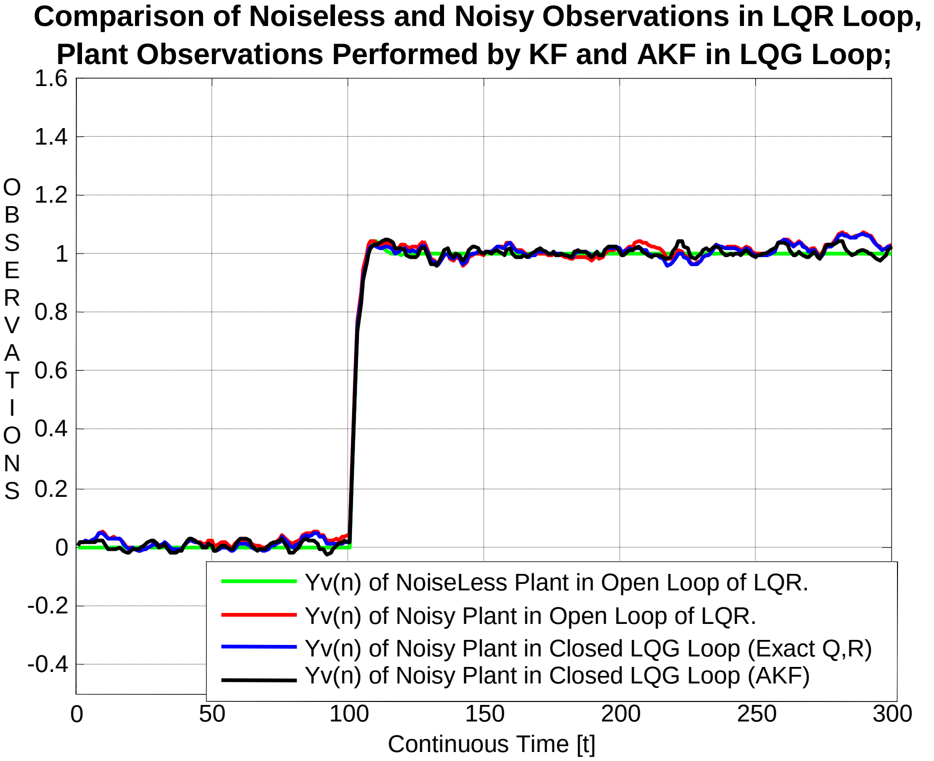


Figure 14.2-4 Observations obtained from different regulators with low power of process noise in plant.

	High power of process noise	Low power of process noise
MSE	$4.53 \, . \, 10^{-4}$ %	$1.31 \, . \, 10^{-4}$ %
RMSE	$2.13 \, . \, 10^{-1}$ %	$1.14 \, . \, 10^{-1}$ %

Table 14.2-1 A comparison between LQG regulators with either KF or AKF.

The table confirms the effectiveness and strong opportunity to use AKF in LQG due to very low MSE and RMSE. Next two tables present the capability of LQG to attenuate the process noise as an uncertainty in states. This uncertainty happens inside of a plant can be only simulated.

10th Order Filter	High Power of Process Noise		Low Power of Process Noise	
Type of employed KF	NNRx in States	NNRy_v in Observations	NNRx in States	NNRy_v in Observations
LQR with AKF	-11.16 dB	-12.41 dB	3.32 dB	-4.12 dB
LQR with Exact Q and R of KF	-11.19 dB	-13.20 dB	-2.64 dB	-1.39 dB

Table 14.2-2 The capability of control system to attenuate a process noise; tenth order.

In Table 14.2-2, the NNR values measured with dB represent the power of an uncertainty in states of plant. The control system in closed-loop, see Figure 14.2-1, provides lower power of an uncertainty than the system of open-loop (Figure 14.2-2) does. This expected fact is presented with negative dB values of NNR.

Let us compare the performance of LQG control closed-loop system with CoKF and AKF. In case of high power of process noise, the regulators with AKF and CoKF provide a little uncertainty in states and observations, see Figure 14.2-3. Additionally, the AKF is capable to cope with high process noise as well as the optimal CoKF. The problem here is that ***Q*** and *R* may be not obtained exactly in practice, and therefore AKF must be used.

In case of low power of process noise, the employed AKF has worse influence to plant than the optimal CoKF a bit, but this fact does not give catastrophic control because the process noise is 100-times lower than in the previous case, see Figure 14.2-4. Thus the observations are not affected at all.

An assessment of a performance of LQG design based regulator and third order filter.

Table 14.2-3 presents the facts that AKF performs very well. So the states and observations have in one or 1.5 dB lower uncertainty than the control system with CoKF has. The same could be said in case of NNRy measured in observations with low power of process noise.

3rd Order Filter	High Power of Process Noise		Low Power of Process Noise	
Type of KF	NNRx in States	NNRy_v in Observations	NNRx in States	NNRy_v in Observations
LQR with AKF	-10.68 dB	-12.08 dB	1.34 dB	-5.34 dB
LQR with Exact Q and R of KF	-9.31 dB	-10.59 dB	-2.20dB	-1.25 dB

Table 14.2-3 The capability of control system to attenuate a process noise; third order.

14.3 An example of multi-state-sensor control system in LQG design

Previous section solved the control problem of a system composed of one sensor. This section deals with LQG based regulator and three sensors employed in the control. Here the LQR gain is taken over the Section 14.2. Additionally, we assume adaptive decentralized Kalman filter-ADKF in multi-state multi-sensor fusion. To specify the control strategy, decentralized structure should be introduced. Our example contains three nodes and every node contains one processor executing program. Thus every node is sensing independent observations $y_{vi}(n)$, where $i = 1, 2, 3$. The model contains three nodes as well as usually three components are controlled such as position, velocity and acceleration. The term "multi-state fusion" has been deduced. By the way, every processor should perform Kalman filtering needed for estimation in LQG problem, and linear-quadratic control in digital-time domain. All provide the same state estimation The Figure 14.3-1 depicts observations produced by regulators.

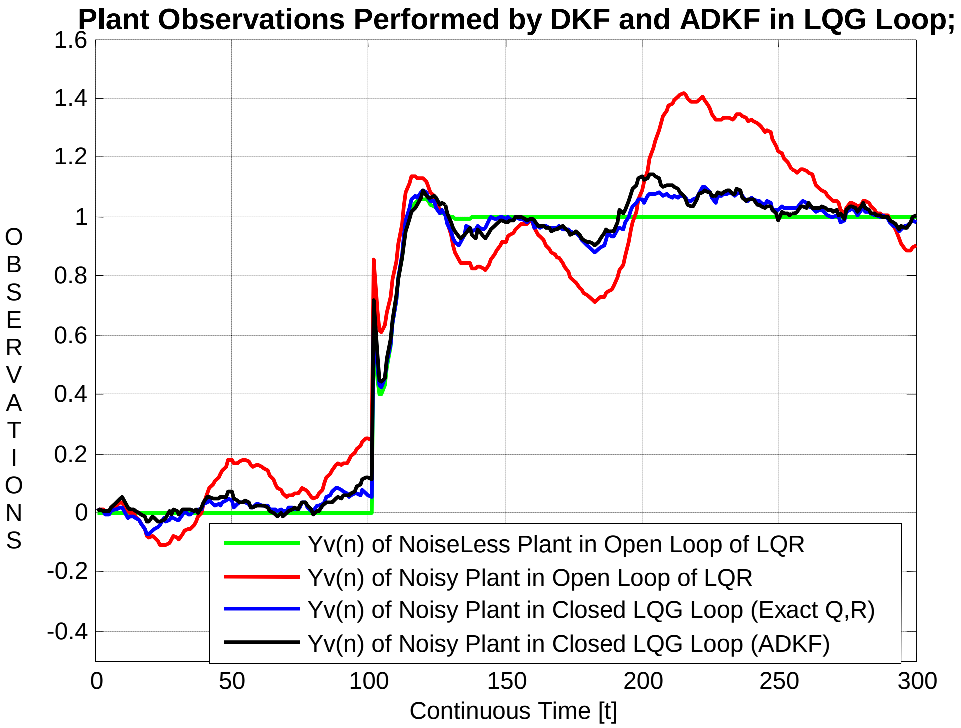


Figure 14.3-1 Observations obtained from different LQG controllers with high power of process noise in plant.

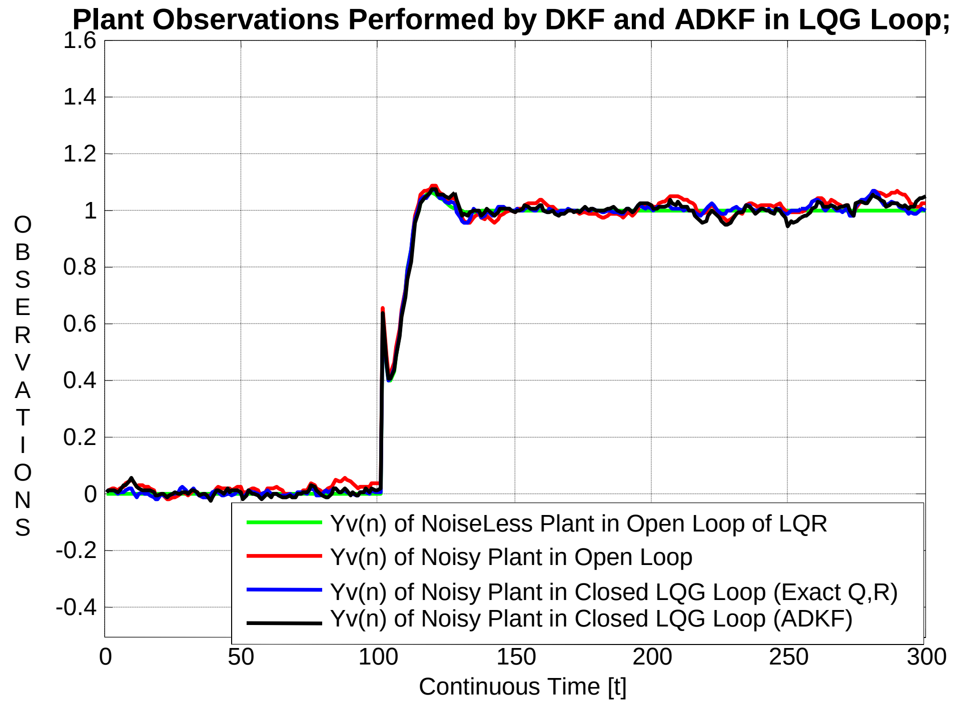


Figure 14.3-2 Observations obtained from different LQG controllers with low power of process noise in plant.

Figure 14.3-3 presents the schema of LQG problem based control with ADKF, but as an alternative the DKF can be used instead. The nodes share their $\boldsymbol{VEI_i}(n)$ and $\boldsymbol{SEI_i}(n)$ information to update their state estimates, where i = 1, 2 and 3. This is SIMO model.

We would like to compare some results of LQG (Table 14.3-1) with three sensors and ADKF toward the LQG (Table 14.2-3) with one sensor and AKF described in Section 14.2. In case of high power of process noise, both regulators provide comparable noisy states of plant and observation $y_{v1}(n)$. In case of low power, our regulator (Table 14.3-1) with three sensors performs in 2.81 dB noiseless controlled state of plant than regulator (Table 14.2-3) with one sensor. The system model equation (12-1) remains been used here too.

3rd Order Filter	**High Power of Process Noise**		**Low Power of Process Noise**	
LQG Closed Loop	**NNRx in States**	**NNRyv in Observations**	**NNRx in States**	**NNRyv in Observations**
LQR with ADKF	-10.35 dB	-10.99 dB	-1.47 dB	-2.88 dB

Table 14.3-1 The capability of complicate control system to attenuate a process noise.

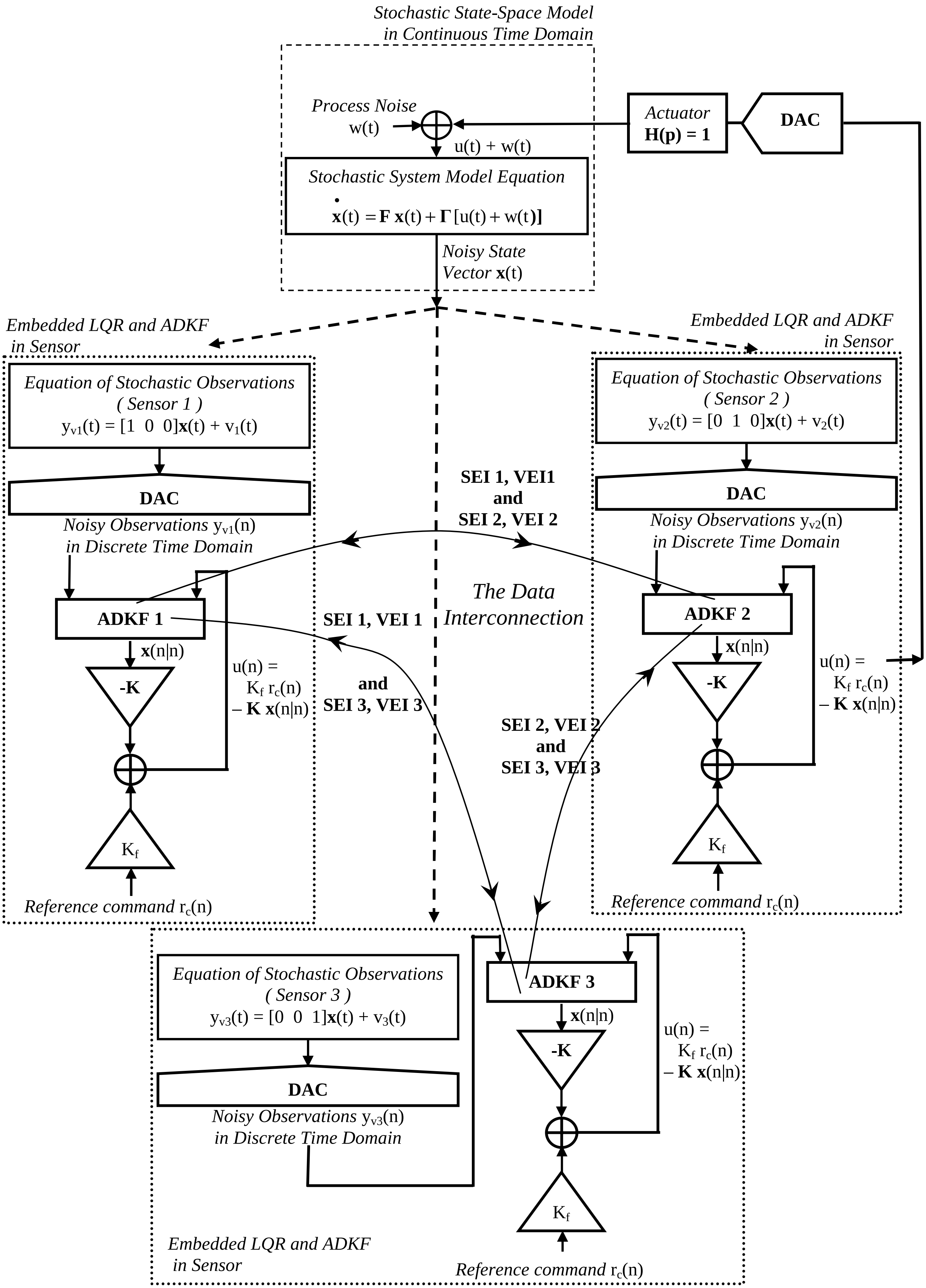

Stochastic State-Space Model
in Continuous Time Domain
Process Noise
w(t)
u(t) + w(t)
Actuator
H(p) = 1
DAC
Stochastic System Model Equation
$\dot{\mathbf{x}}(t) = \mathbf{F}\,\mathbf{x}(t) + \mathbf{\Gamma}\,[u(t) + w(t)]$
Noisy State
Vector $\mathbf{x}(t)$
Embedded LQR and ADKF
in Sensor
Equation of Stochastic Observations
(Sensor 1)
$y_{v1}(t) = [1\ \ 0\ \ 0]\mathbf{x}(t) + v_1(t)$
DAC
Noisy Observations $y_{v1}(n)$
in Discrete Time Domain
ADKF 1
$\mathbf{x}(n|n)$
-K
$u(n) = K_f\, r_c(n) - \mathbf{K}\,\mathbf{x}(n|n)$
K_f
Reference command $r_c(n)$
SEI 1, VEI1
and
SEI 2, VEI 2
The Data
Interconnection
SEI 1, VEI 1
and
SEI 3, VEI 3
SEI 2, VEI 2
and
SEI 3, VEI 3
Embedded LQR and ADKF
in Sensor
Equation of Stochastic Observations
(Sensor 2)
$y_{v2}(t) = [0\ \ 1\ \ 0]\mathbf{x}(t) + v_2(t)$
DAC
Noisy Observations $y_{v2}(n)$
in Discrete Time Domain
ADKF 2
$\mathbf{x}(n|n)$
-K
$u(n) = K_f\, r_c(n) - \mathbf{K}\,\mathbf{x}(n|n)$
K_f
Reference command $r_c(n)$
Equation of Stochastic Observations
(Sensor 3)
$y_{v3}(t) = [0\ \ 0\ \ 1]\mathbf{x}(t) + v_3(t)$
DAC
Noisy Observations $y_{v3}(n)$
in Discrete Time Domain
ADKF 3
$\mathbf{x}(n|n)$
-K
$u(n) = K_f\, r_c(n) - \mathbf{K}\,\mathbf{x}(n|n)$
K_f
Embedded LQR and ADKF
in Sensor
Reference command $r_c(n)$

Figure 14.3-3 Schema of LQG problem based control in multi-sensor multi-state system.

14.4 Conclusions

Below we apply the last question asked in section 1.1 to this conclusion:
"Demonstrate a functionality of an optimal regulator in Linear Quadratic Gaussian, LQG, problem, so that a single/multi dimensional output of a linear system takes control in backward closed loop. In addition to linear system, the (adaptive) Kalman filter should be investigated."

The Unit 14 demonstrates technique of LQR closed-loop problem as an approach to simple exercises. This technique controls the step response of system to satisfy control criteria.
Section 14.1 deals with similar exercises with considered nonlinear actuator. The advantage of this control solution is knowledge on actuator. A different performance occurs between LQR presented in last two sections.
In sections 14.2 and 14.3 the system controlled in closed-loop provides lower uncertainty than in open-loop control. All measurements are performed in states and observations and AKF is employed in LQG. Section 14.3 deals with LQG problem based regulator and three sensors fully employed in the control. There the adaptive decentralized Kalman filter-ADKF is employed in multi-sensor multi-state fusion additionally. The states provided to LQG + ADKF is nearly same with in LQG + DKF.

Considered compensator has very high order equal to the order of plant. This means that it has too many parameters to be conveniently gain scheduled. It might be possible to use reduced-order compensator principle component analysis-PCA as one of many other techniques.

15 MAIN SUMMARY AND CONCLUSIONS

This manuscript is composed of two main parts as well as Kalman-Filter techniques, MATLAB simulations and tests.

The first part employs Kalman-Filter technique with one sensor, centralized and decentralized Kalman-Filter technique and technique of adaptive Kalman-Filter. Unit 2 to 5 are related to those techniques. The second part deals with some experiments of conventional Kalman-Filter with one sensor so-called CoKF; centralized Kalman-Filter with many sensors so-called CKF; decentralized-centralized Kalman-Filter comparison – DKF. Then the followings are associated with experiments in problematic of bias, broken node, drift effect on state estimation in DKF, algorithm of fault detection and isolation in sensors, algorithm and exercising on adaptive fuzzy logic decentralized Kalman-Filter, LQR and LQG.

The section 1.1 gives seven questions to be answered one by one below.

We apply the first question to summary:

"Demonstrate state estimation identity of decentralized Kalman filter, DKF, and centralized Kalman filter, CKF, in an experimental simulation study assuming uniform filter conditions. What are differences?"

The states dealt with both estimators appear nearly identical. This summary refers to Unit 8.

We apply the second question to summary:

"Measure a mean square error, MSE, of state estimation in simulated DKF model that contains a sensor been affected by a bias with uniform filter conditions. Simulations have to be performed with total numbers of sensors such as two and five. How does the MSE depend on the total number of sensors? Tests need to be performed, namely with:

- constant bias,
- linearly increased bias."

The relation between mean square error and the total number of sensors appeared been valid same for both bias types. This relation is explained by two and five sensors, where $\sqrt{\mathrm{MSE}_{2\mathrm{DKF}} / \mathrm{MSE}_{5\mathrm{DKF}}}$ is equal to number of sensors ratio given by $\frac{5}{2}$. The $\mathrm{MSE}_{2\mathrm{DKF}}$ is the mean square error of state estimation in DKF model composed by two sensors, etc. This summary refers to Unit 9.

We apply the third question to summary:

"Measure the MSE of state estimation in simulated DKF model that contains a broken sensor with uniform filter conditions. Simulations have to be performed with two and five total numbers of sensors. How does the MSE depend on the total number of sensors?"

The DKF model is composed of two and five sensors, where only one of them is broken.

Ratio $\sqrt{\mathrm{MSE}_{2\mathrm{DKF}} / \mathrm{MSE}_{5\mathrm{DKF}}}$ is equal to 2.496 and 2.501 assuming high and low power of process noise, respectively. Both values (2.496 and 2.501) are numerically near the expected value 2.5. The $\mathrm{MSE}_{2\mathrm{DKF}}$ is the mean square error of state estimation in DKF model composed by two sensors, etc. This summary refers to Unit 10.

We apply the fourth question to summary:

"Measure the MSE of state estimation in DKF model that contains a sensor affected by a drift with uniform filter conditions. Simulations have to be performed by using one, two, and five total numbers of sensors. How does the MSE depend on the total number of sensors?"

As an approximation, the $\sqrt{\mathrm{MSE}}$ is linearly proportional to the used total number of sensors. For example in our case of 2 sensors model, the estimator gives four times lower MSE of state estimation than DKF with one sensor. Five sensors DKF performs with 25-times lower MSE than one sensor DKF. Exponential drift is assumed. This summary refers to Unit 11.

We apply the fifth question to summary:

“Find out a fault detection and isolation, FDI, algorithm been possibly useable for DKF model. This model has to be tested with a minor number of affected sensors by exponentially descending drifts. What is a performance of this DKF model with applied FDI algorithm, when:

- one sensor is affected by the drift, and additionally two sensors are correct;
- two sensors are affected by two miscellaneous drifts, plus one correct sensor?”

The major requirement on FDI algorithm is the availability to multiple observations. It was shown on how to apply FDI algorithm to the problem of DKF model with faults in sensors. The FDI algorithm was already found out. Firstly the question related to one affected sensor is to be answered below.

Our simulation results shown that the DKF model with FDI algorithm yields 75% reliability of FDI performance insomuch that a pure DKF model without FDI provides exactly 0% reliability. The FDI reliability is not meant to be state estimation accuracy, although its relative MSE is about 0.01. In case of high process noise, the simulation results gave about 70% reliability of FDI algorithm. The 5 percentage aggravation is caused by 100-times higher power of process noise, where cross-correlation function used in FDI system is disturbed by that.

Having two affected sensors by exponential drifts, the FDI reliability was 84%. By considered 100-times higher power of process noise, the reliability was reduced down to 54%, but relative MSE of state estimation was about 0.11. This is the FDI imperfection with major number of corrupted sensors.

The DKF models use multi-state, multi-sensor system, meaning fusion with different observations quality. Time behaviour of $u(n)$ is sinus as a still oscillated signal rather used than a less problematic step pulse. This summary refers to Unit 12.

We apply the sixth question to summary:

“Test an algorithm of adaptive Kalman filter, AKF, supported by fuzzy logic to meet the Kalman filter functionality, by definition of an optimal state estimation in Kalman filtering, with a strategy of an adapted process noise covariance $\boldsymbol{Q}$ matrix and observation R_1 noise variance. Suppose both prior $\boldsymbol{Q}$ and R_1 are unknown. The reasonable effort should be made to additionally estimate the covariance matrices when the noise streams are unknown. So that, the AKF extends KF functionality to meet the practical needs when changing from traditional KF."

To exam the AKF with high power of process noise, a bit better state estimation accuracy occasionally occurs than the conventional Kalman filter provides. Considering lower power of process noise than the power of sensor noise, the $w(n)$ and $v_1(n)$ are very little and can not worse the state estimation at all, especially in control. The adaptive algorithm is considerable, otherwise an absence of that inappropriate Q and R_1 setting may cause a practical divergence of estimator, but this effect has been never found in AKF. This summary refers to Unit 13.

We apply the seventh question to summary:

"Demonstrate a functionality of an optimal regulator in Linear Quadratic Gaussian, LQG, problem, so that a single/multi dimensional output of a linear system takes control in backward closed loop. In addition to linear system, the (adaptive) Kalman filter should be investigated."

The technique of LQR in closed-loop controls the step response of system to satisfy control criteria.

Section 14.1 deals with some exercises with nonlinear actuator. A slight performance occurs between LQR with linear/nonlinear actuator. Of course it is depending on the type of nonlinearity to be presented in specific control.

The LQG has a capability to attenuate system's uncertainty in controlled LQG closed-loop. This is of course a by-product. The main product is good states in plant that can be based on comparison of LQG and LQR.

In addition to ADKF employed in multi-sensor multi-state fusion, the states provided to LQG + ADKF is nearly same with in LQG + DKF. This summary refers to Unit 14.

REFERENCES

[1] **GOODWIN Clifford G., and SIN K. S. (1984). “Adaptive Filtering Prediction and Control,” PRENTICE-HALL, INC., Englewood Cliffs, New Jersey 07632.**

[2] **FRIEDLAND B. (1996). “Advanced Control System Design,” PRENTICE-HALL, INC., A Simon & Schuster Company Englewood Cliffs, New Jersey 07632**

[3] **GELB A., KASPER J. F., NASH R. A., PRICE C. F., and SUTHERLAND A. A., (1999). “Applied Optimal Estimation,” Massachusetts Institute of Technology Cambridge, Massachusetts, and London, England, Library of Congress Catalog Card Number: 741604.**

[4] **MOHINDER S. GREWAL, and ANGUS P. ANDREWS, (1993). “KALMAN FILTERING Theory and Practice,” PRENTICE-HALL, INC., A Simon & Schuster Company Englewood Cliffs, New Jersey 07632.**

[5] **FRANK L. LEWIS, and VASSILIS L. SYRMOS, (1993). “OPTIMAL CONTROL,” A Wiley-Interscience Publication, A John Wiley & Sons Inc., New York, Chichester, Brisbane, Toronto, Singapore.**

[6] **ISERMANN R. (Rolf), (2003). “Mechatronic Systems : fundamentals,” 1.Mechatronics and 2.Microelectromechanical systems, British Library Cataloguing in Publication Data.**

[7] **MATLAB Release 12 Help.**

[8] **VERHAEGEN M. H., (March 1989). “Improved understanding of the loss-of-symmetry phenomenon in the conventional Kalman filter,” Automatic Control, IEEE Transactions, Vol.34, Iss.3, pp. 331-333.**

[9] RAO B. S., and DURRANT-WHYTE H. F., (September 1991). “Fully decentralized algorithm for multisensor Kalman filtering,” IEE PROCEEDINGS-D, Vol. 138, No. 5, pp. 413-420.
http://portal.acm.org/citation.cfm?id=942356&dl=ACM&coll=portal

[10] PAO L. Y., and BALTZ N. T. (June 1999). “Control of Sensor Information in Distributed Multisensor Systems,” Proc. American Control Conference, San Diego, CA,pp. 2397-2401.

[11] DURRANT-WHYTE H. F., and LEONARD J. J., (13-18 May 1990). “Toward a fully decentralized architecture for multi-sensor data fusion,” Robotics and Automation, 1990.Proceedings., IEEE International Conference, Vol.2, pp. 1331 – 1336.
http://ieeexplore.ieee.org/iel5/484/3534/00126185.pdf?tp=&arnumber=126185&isnumber=3534

[12] DURRANT-WHYTE H. F., RAO B. Y. S., and HU H., (4 Feb 1991). “Toward a fully decentralized architecture for multi-sensor data fusion,” Principles and Applications of Data Fusion, IEE Colloquium, pp.2/1 - 2/4.
http://ieeexplore.ieee.org/iel3/1743/4555/00180989.pdf?tp=&arnumber=180989&isnumber=4555

[13] DURRANT-WHYTE H. F., (19 Feb 1991). “Elements of sensor fusion,” Intelligent Control, IEE Colloquium, pp.5/1 - 5/2.
http://ieeexplore.ieee.org/iel3/1731/4566/00181126.pdf?tp=&arnumber=181126&isnumber=4566

[14] DAILEY DANIEL J., HARN PATRICIA, and LIN PO - JUNG, (April 1996). “ITS Data Fusion,” Final Research Report T9903, University of Washington Seattle, Washington State Department of Transportation Technical Monitor.
http://www.its.washington.edu/pubs/fusion_report.pdf

[15] WEI M., and SCHWARTZ K. P., (20-23 March 1990). “Testing a decentralized filter for GPS/INS integration,” Position Location and Navigation Symposium, 1990. Record. 'The 1990's - A Decade of Excellence in the Navigation Sciences'. IEEE PLANS '90., IEEE , pp. 429 – 435.
http://ieeexplore.ieee.org/iel2/114/2376/00066210.pdf?tp=&arnumber=66210&isnumber=2376

[16] **STROBEL N., SPORS S. and RABENSTEIN R., (5-9 June 2000). "Joint audio-video object localization using a recursive multi-state multi-sensor estimator," Acoustics, Speech, and Signal Processing, 2000. ICASSP '00. Proceedings. 2000 IEEE International Conference, Vol.6, pp.2397 - 2400 vol.4.**
http://ieeexplore.ieee.org/iel5/6939/18643/00859324.pdf?tp=&arnumber=859324&isnumber=18643

[17] **TAE-GYOO LEE., (December 2003). "Centralized Kalman Filter with Adaptive Measurement Fusion: its Application to a GPS/SDINS Integration System with an Additional Sensor," International Journal of Control, Automation, and Systems Vol. 1, No. 4.**

[18] **LOEBIS D., CHUDLEY J., and SUTTON R. (1 Aprile 2004). "A fuzzy Kalman filter optimized using a multi-objective genetic algorithm for enhanced autonomous underwater vehicle navigation," Proceedings of the I MECH E Part M Journal of Engineering for the Maritime Envi, vol. 218, no. 1, pp. 53-69(17).**

[19] **RAY ASOK, and LUCK ROGELIO (Aprile 7-12 1991). "An Introduction to Sensor Signal Validation in Redundant Measurement Systems," The 1991 IEEE International Conference on Robotics and Automation, Hyatt Regency in Sacramento, California, pp. 44-49.**

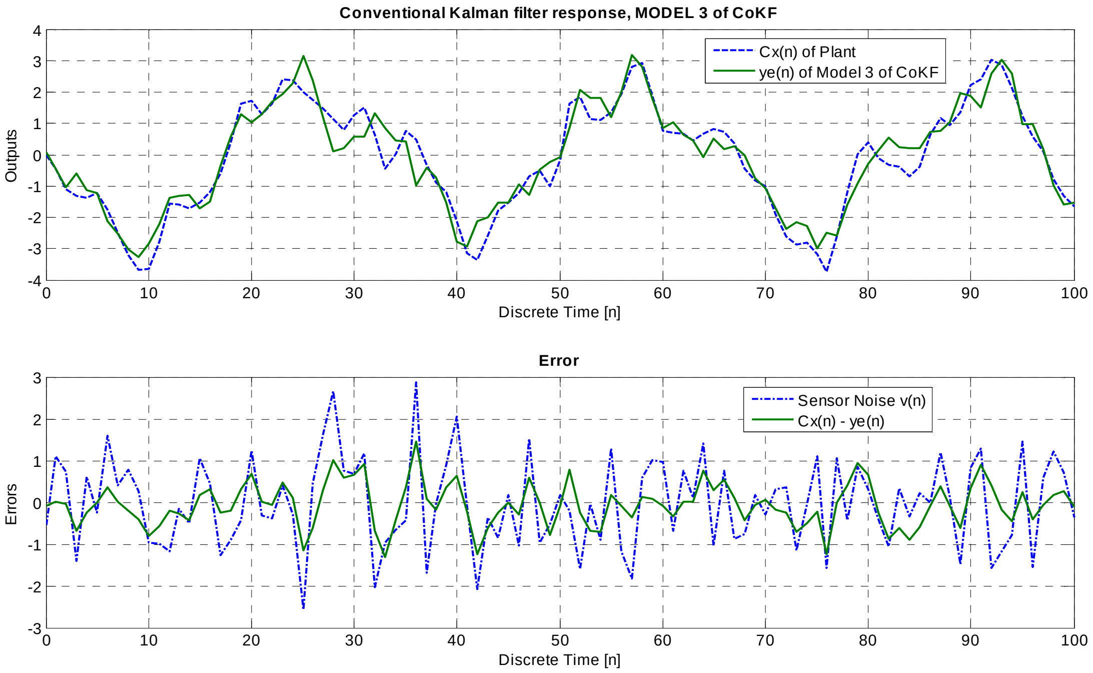
[20] **FITZGERALG R.J. (1997). "Divergence of the Kalman Filter," IEEE Transaction on Automatic Control, vol. AC-16, no. 6, pp.736-747.**

[21] **ROGÉRIO BASTOS QUIRINO and CELSO PASCOLI BOTTURA (January – April 2001). "An approach for distributed Kalman filtering," SBA Control and Automation, vol. 12, no. 01.**

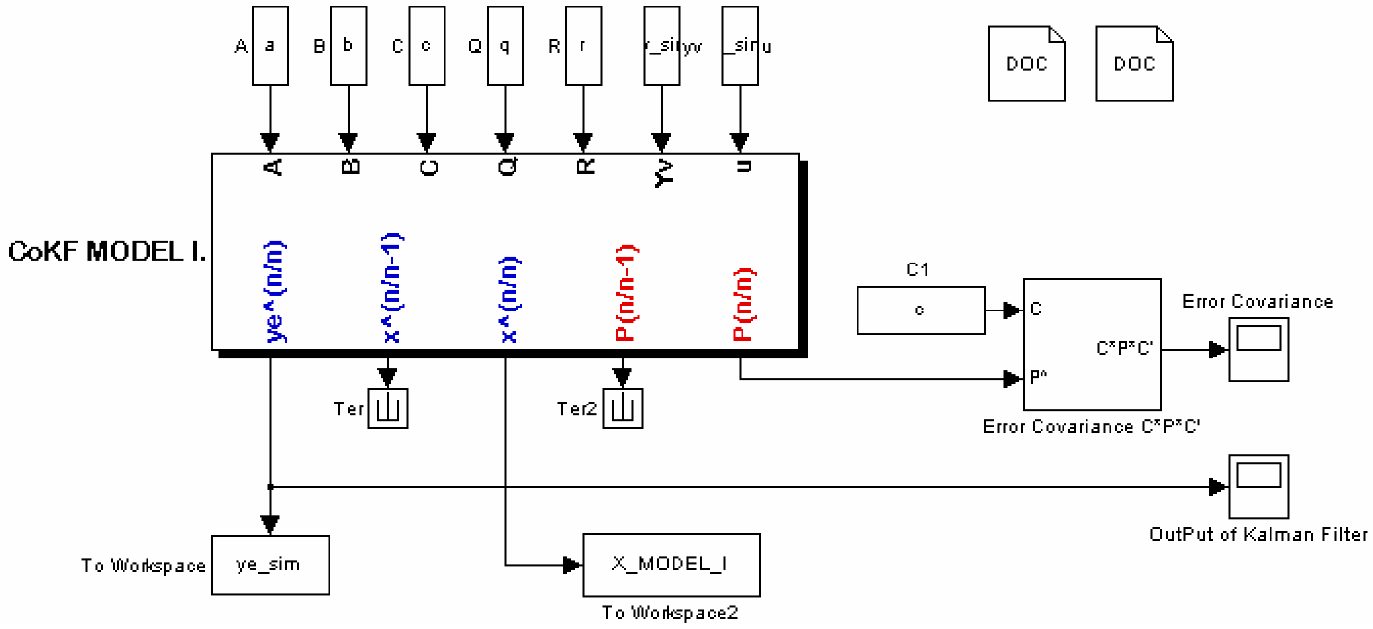
APPENDIX A

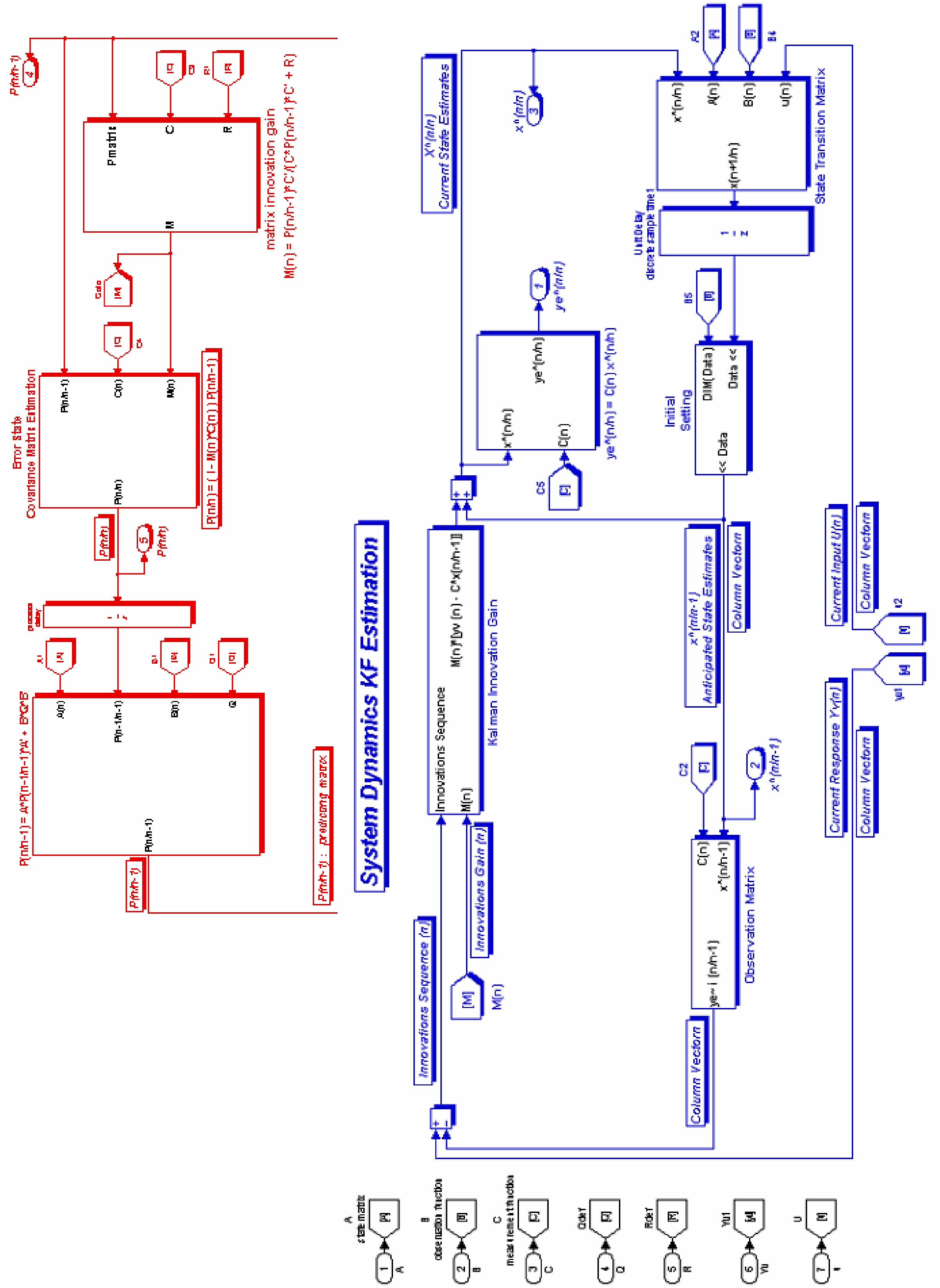
A A BRIEF REVIEW OF SOME RESULTS FROM CoKF TECHNIQUES

A.1 Model 3 of KF Result (referring to Unit 6)

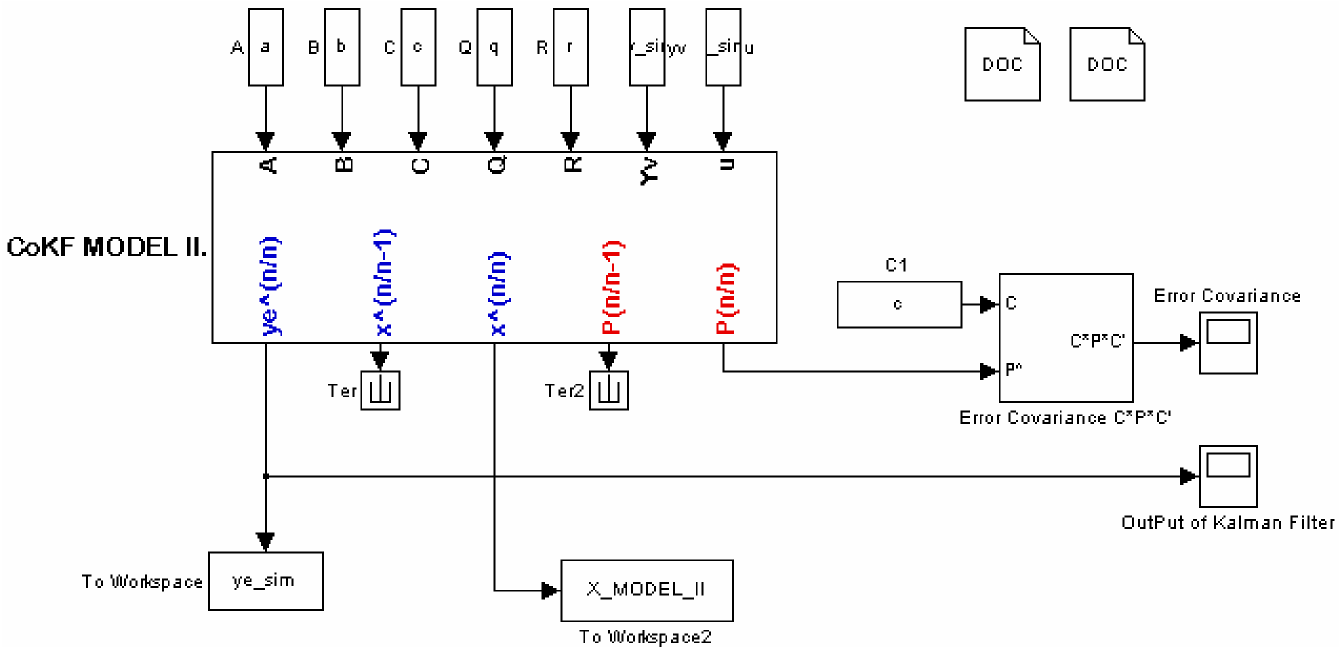


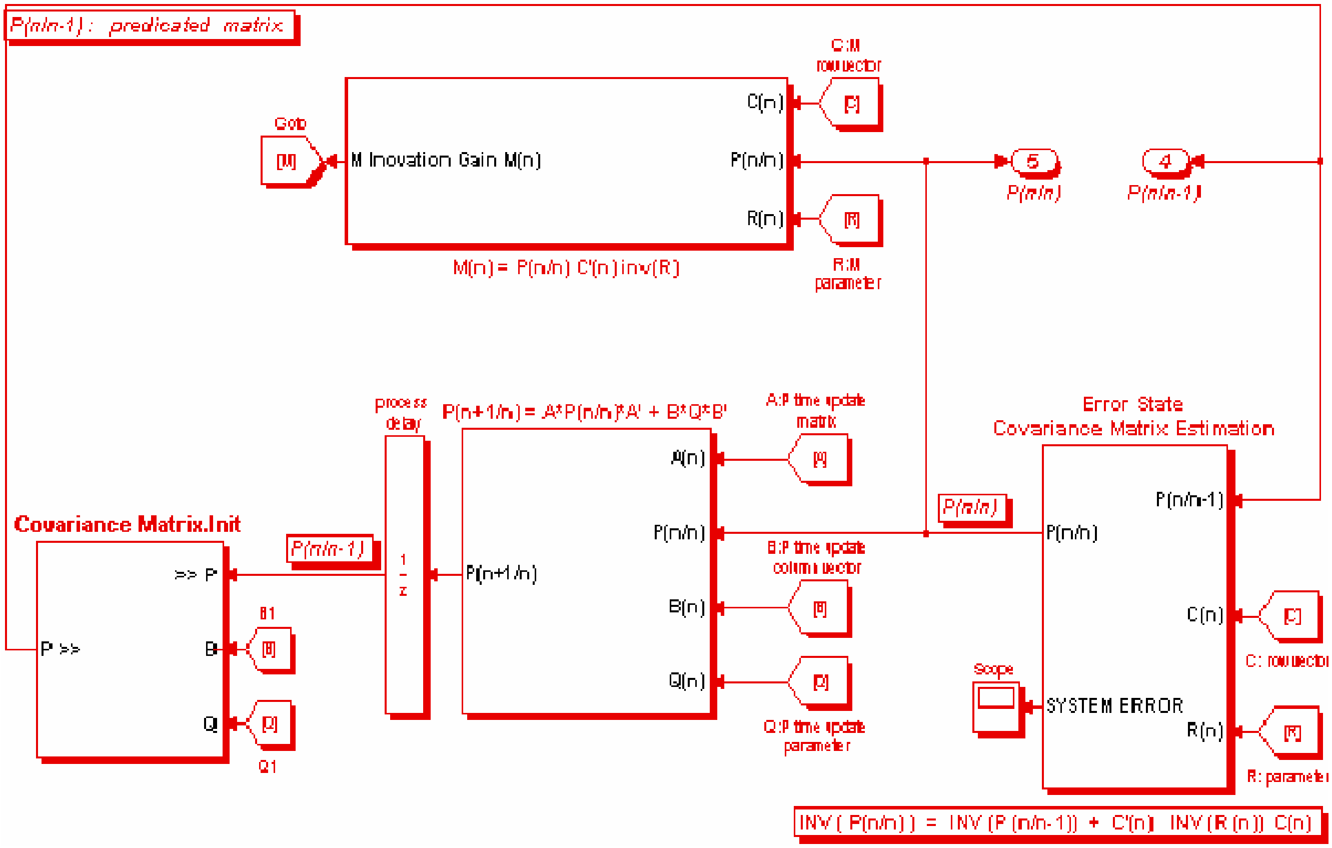
A.2 Simulink Block of Model 1 of KF (referring to Table 2-1 in Chapter 2)



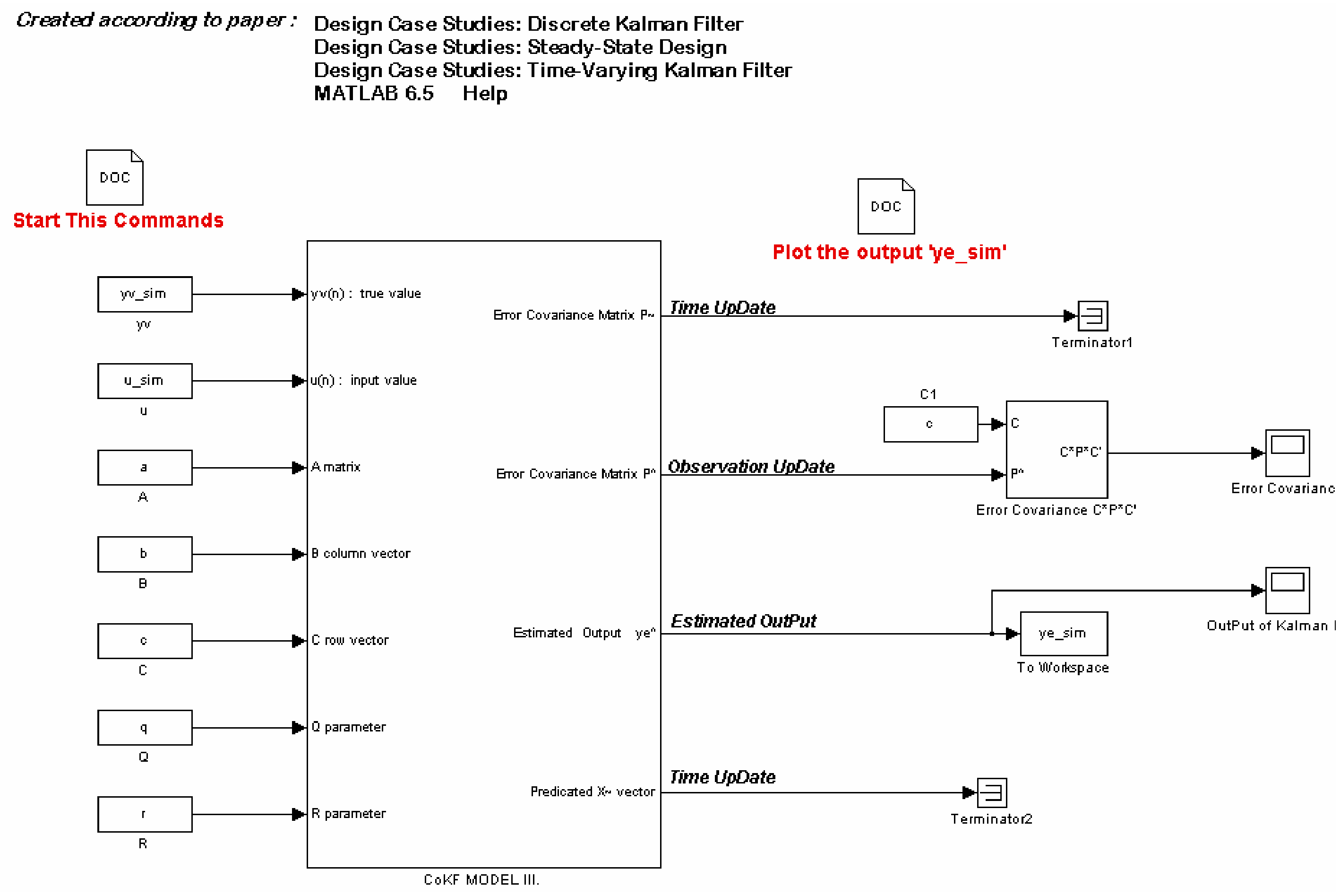

System Dynamics KF Estimation
P(n/n-1)
Pmatrix
C
R
M
matrix innovation gain
M(n) = P(n/n-1)*C'/(C*P(n/n-1)*C' + R)
Gain
Error state
Covariance Matrix Estimation
P(n/n-1)
C(n)
M(n)
P(n/n)
P(n/n) = (I - M(n)*C(n)) P(n/n-1)
P(n/n)
5
P(n/n)
unit delay
1/z
P(n/n-1) = A*P(n-1/n-1)*A' + B*Q*B'
A(n)
P(n-1/n-1)
B(n)
Q
P(n/n-1)
P(n/n-1)
P(n/n-1) : predicting matrix
X^(n/n)
Current State Estimates
x^(n/n)
3
1
ye^(n/n)
x^(n/n)
ye^(n/n)
C(n)
ye^(n/n) = C(n) x^(n/n)
C5
Innovations Sequence
M(n)*[yv(n) - C*x(n/n-1)]
M(n)
Kalman Innovation Gain
Innovations Gain (n)
Innovations Sequence (n)
M(n)
Unit Delay
discrete sample time1
x^(n/n)
A(n)
B(n)
u(n)
x(n+1/n)
State Transition Matrix
A2
B4
B5
Initial Setting
DIM(Data)
Data <<
<< Data
x^(n/n-1)
Anticipated State Estimates
Column Vectorn
C2
2
x^(n/n-1)
C(n)
x^(n/n-1)
ye~ (n/n-1)
Observation Matrix
Column Vectorn
Current Input U(n)
Column Vectorn
Current Response Y(n)
Column Vectorn
f2
yel
1 A
A
state matrix
2 B
B
observation function
3 C
C
measurement function
4 Q
QdeY
5 R
RdeY
6 Yu
Yu1
7 u
U

A.3 Simulink Block of Model 2 of KF (referring to Table 2-2 in Unit 2)



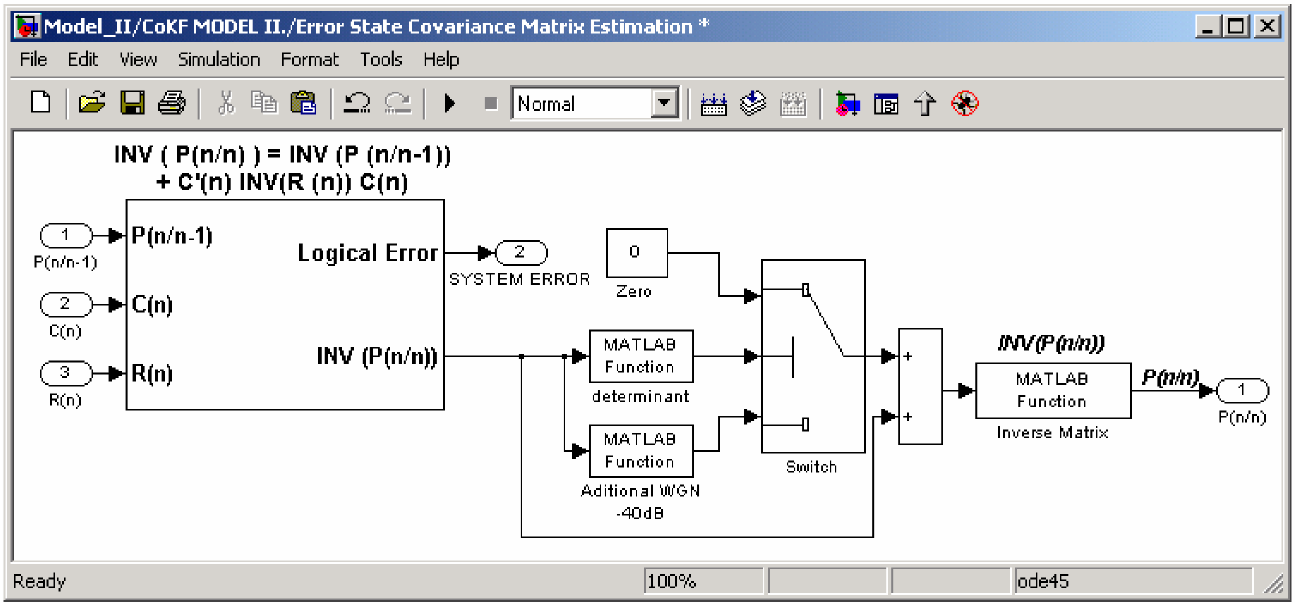


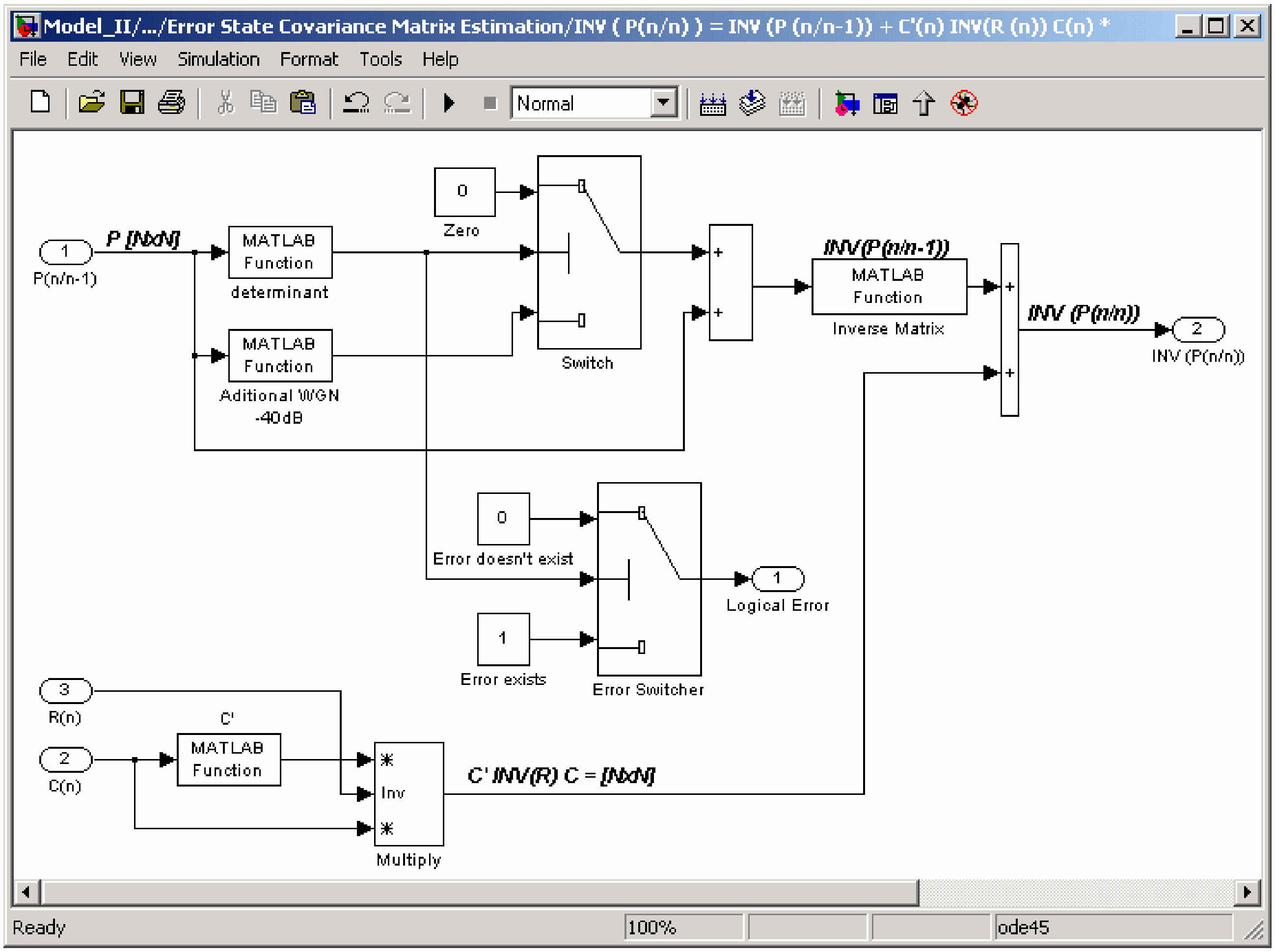
A.4 Simulink Block of Model 3 of KF (referring to Table 2-3 in Unit 2)



Created according to paper : Design Case Studies: Discrete Kalman Filter
Design Case Studies: Steady-State Design
Design Case Studies: Time-Varying Kalman Filter
MATLAB 6.5 Help

A.5 Improvement of CoKF in Simulink MATLAB Program (referring to the flowchart of Figure 2.1-1)

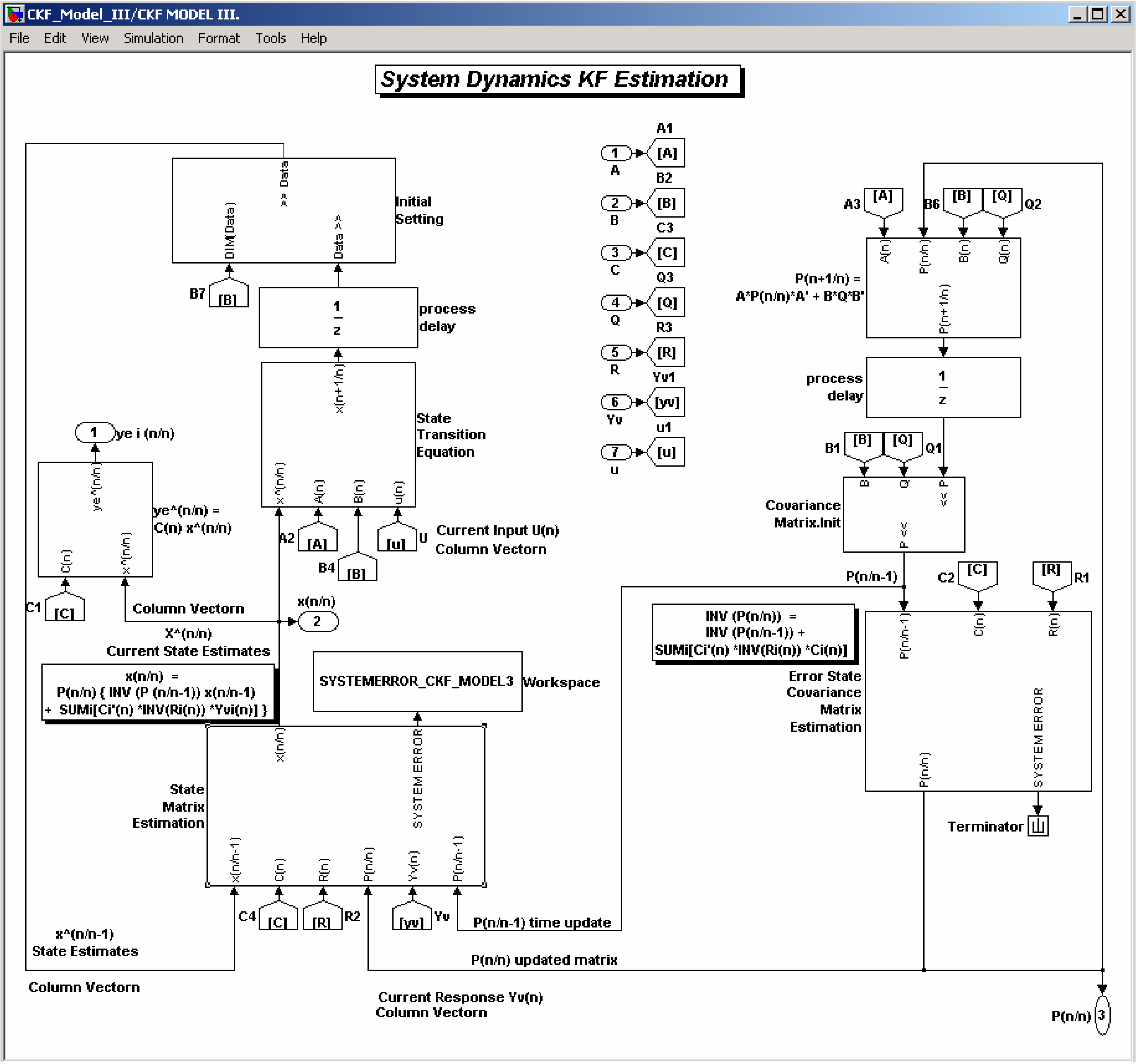




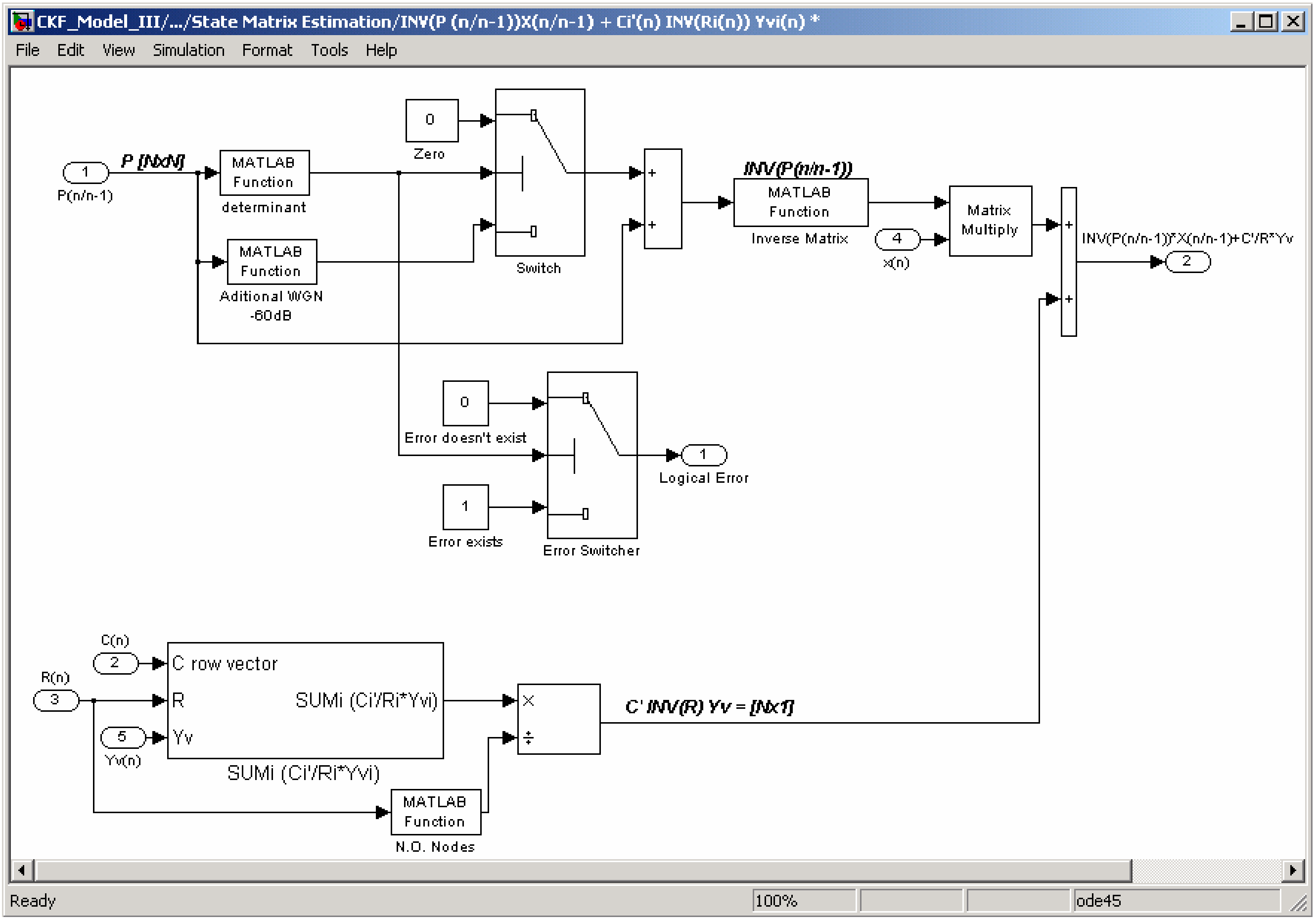
APPENDIX B

B A BRIEF REVIEW OF SOME RESULTS FROM CKF TECHNIQUES

B.1 Inner Block Connection of Fully Centralized KF - Model 3 of CKF (referring to Table 3-3 in Unit 3)



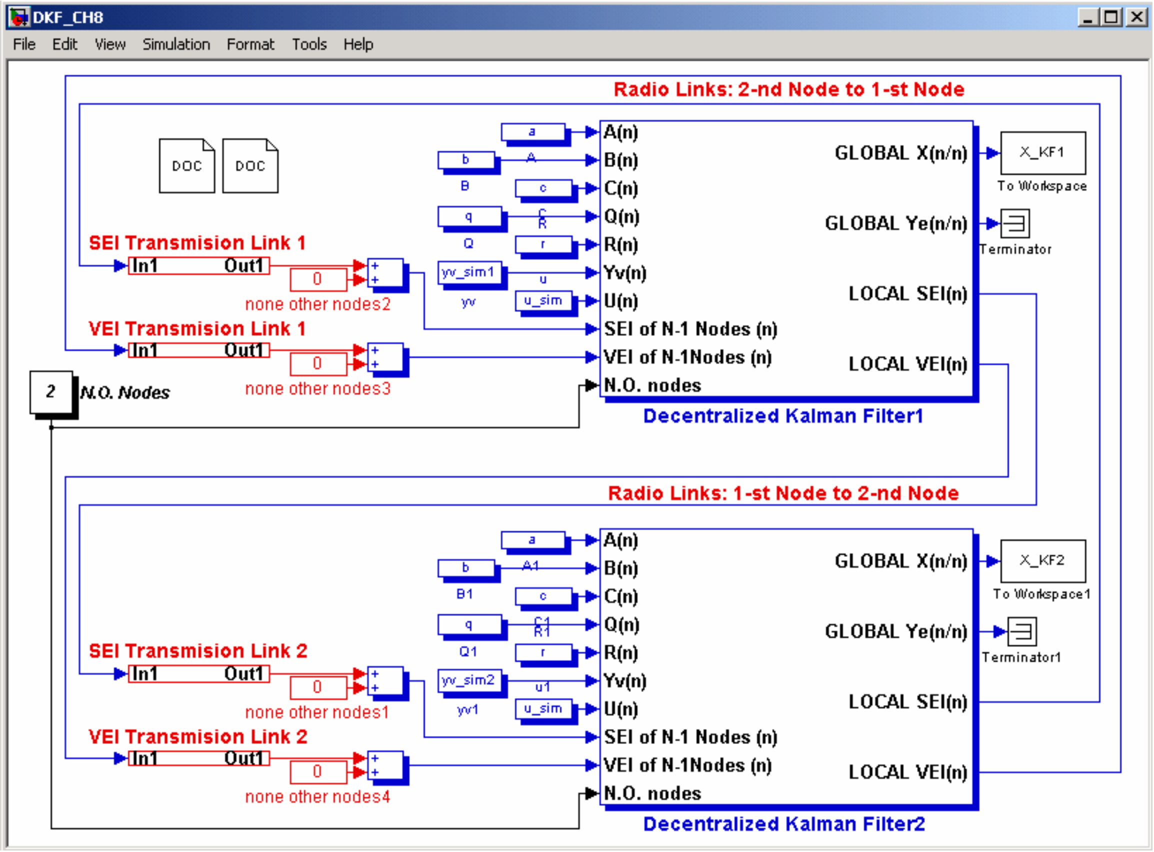
B.2 Improvement of Model 3 of CKF in Simulink MATLAB Program (referring to the treated equation (3-21) in flowchart of Figure 3.1-1, Unit 3)



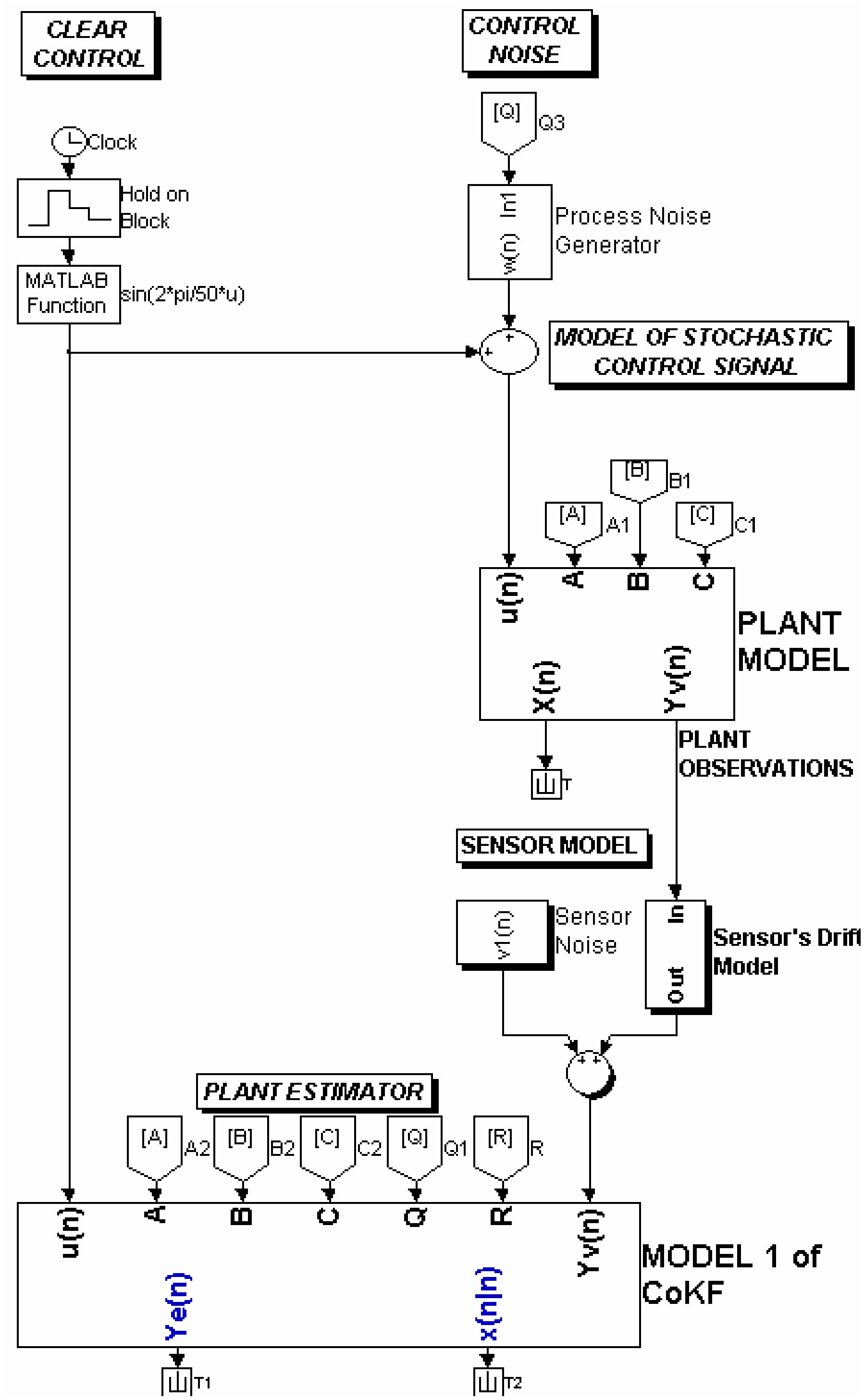
APPENDIX C

C A BRIEF REVIEW OF SOME TESTS AND MODELS

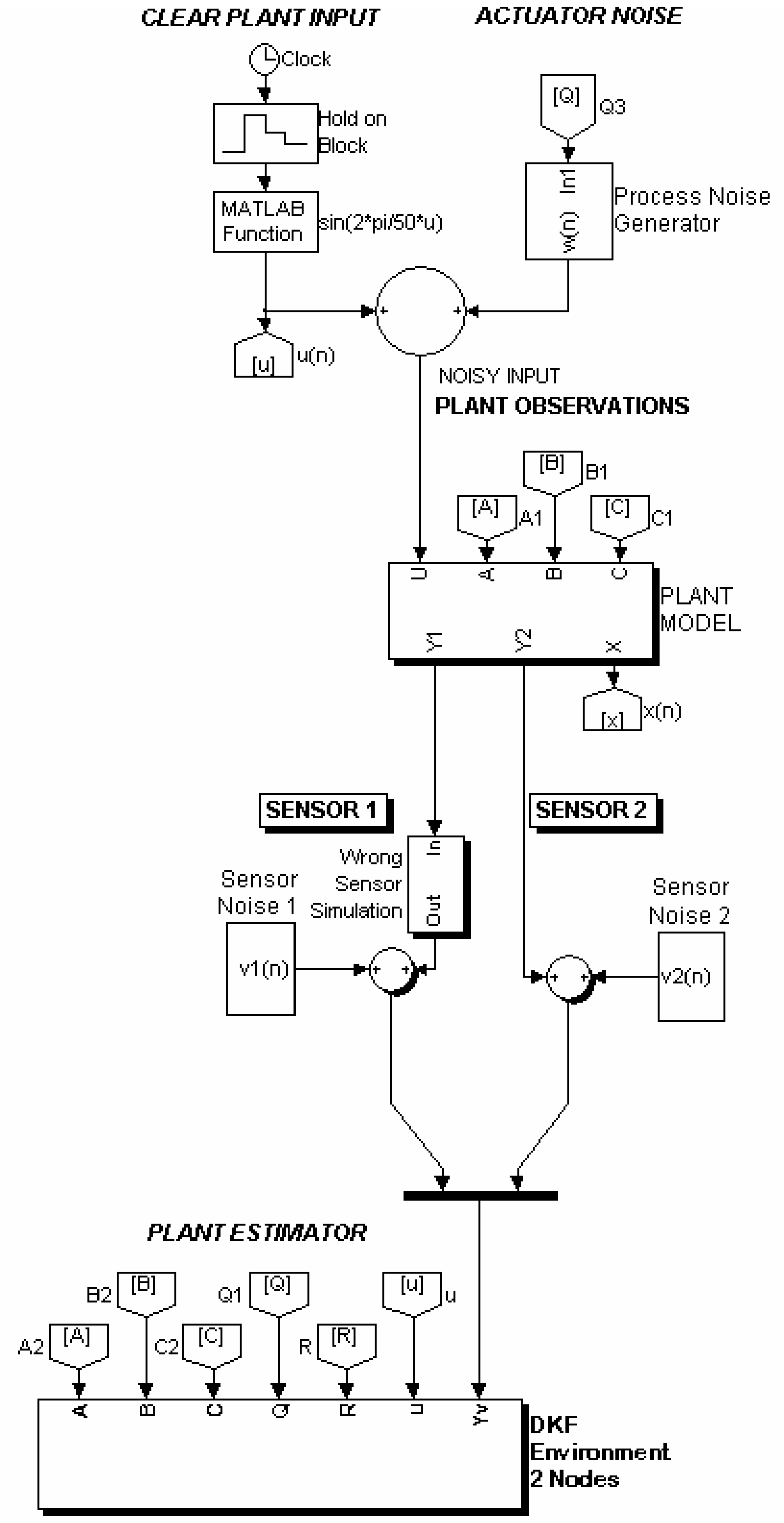
C.1 Two Nodes DKF model (referring to the tests in Unit 8)



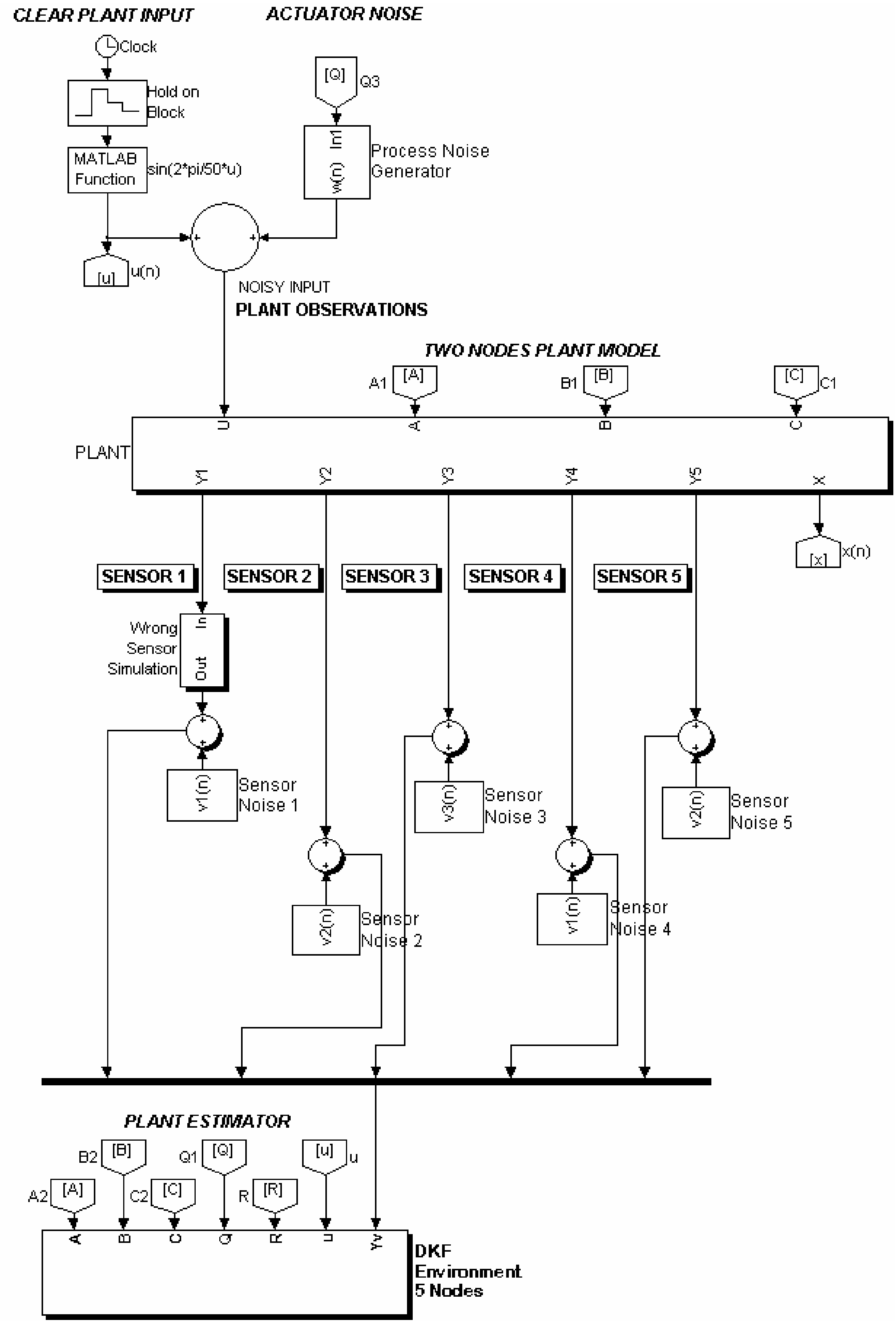
C.2 Test of One Node CoKF model (referring to the Unit 11.1)



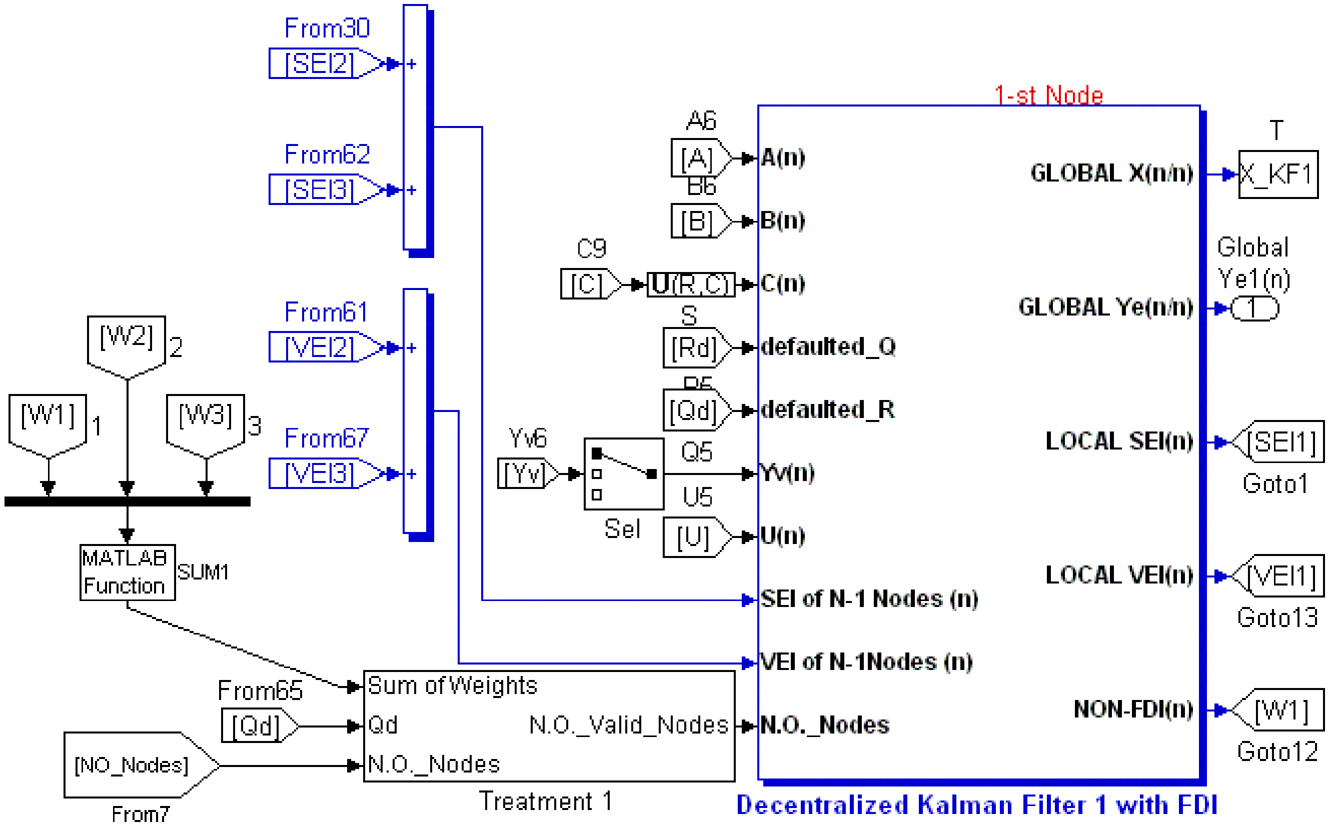
C.3 Test of Two Nodes DKF model (referring to the Units 9.1, 9.3, 10.1, and 11.2)



C.4 Test of Five Nodes DKF model (referring to the Units 9.2, 9.4, 10.2, and 11.3)



C.5 First Node Block Connection of DKF with FDI shown in Simulink MATLAB Program


From30
[SEI2]
From62
[SEI3]
From61
[VEI2]
From67
[VEI3]
[W2]
2
[W1]
1
[W3]
3
MATLAB Function
SUM1
From65
[Qd]
[NO_Nodes]
From7
Sum of Weights
Qd
N.O._Valid_Nodes
N.O._Nodes
Treatment 1
A6
[A]
B6
[B]
C9
[C]
U(R,C)
S
[Rd]
R5
[Qd]
Yv6
[Yv]
Sel
Q5
U5
[U]
1-st Node
A(n)
B(n)
C(n)
defaulted_Q
defaulted_R
Yv(n)
U(n)
SEI of N-1 Nodes (n)
VEI of N-1Nodes (n)
N.O._Nodes
GLOBAL X(n/n)
GLOBAL Ye(n/n)
LOCAL SEI(n)
LOCAL VEI(n)
NON-FDI(n)
Decentralized Kalman Filter 1 with FDI
T
X_KF1
Global Ye1(n)
1
[SEI1]
Goto1
[VEI1]
Goto13
[W1]
Goto12