Algorithm for recursive Frisch scheme ientification and errors-in-variables filtering

Linden, Jens

Award date:
2008

Awarding institution:
Coventry University

Algorithms for recursive Frisch scheme identification and errors-in-variables filtering

Jens G. Linden

Mathematisch-Technischer Assistent/Informatik,
Diplom Mathematiker (FH), MSc Control Engineering

August 2008

A thesis submitted in partial fulfilment of the University's requirements
for the degree of Doctor of Philosophy.

Control Theory and Applications Centre

Coventry University

Abstract

This thesis deals with the development of algorithms for recursive estimation within the errors-in-variables framework. Within this context attention is focused on two major threads of research: Recursive system identification based on the Frisch scheme and the extension and application of errors-in-variables Kalman filtering techniques.

In the first thread, recursive algorithms for the approximate update of the estimates obtained via the Frisch scheme, which makes use of the Yule-Walker model selection criterion, are developed for the case of white measurement noise. Gradient-based techniques are utilised to update the Frisch scheme equations, which involve the minimisation of the model selection criterion as well as the solution of an eigenvalue problem, in a recursive manner. The computational complexity of the resulting algorithms is critically analysed and, by introducing additional approximations, fast recursive Frisch scheme algorithms are developed, which reduce the computational complexity from cubic to quadratic order. In addition, it is investigated how the singularity condition within the Frisch scheme is affected when the estimates are computed recursively. Whilst this first group of recursive Frisch scheme algorithms is developed directly from the offline Frisch scheme equations, it is also possible to interpret the Frisch scheme within an extended bias compensating least squares framework. Consequently, the development of recursive algorithms, which update the estimate obtained from the extended bias compensated least squares technique, is considered. These algorithms make use of the bilinear parametrisation principle or, alternatively, the variable projection method. Finally, two recursive Frisch scheme algorithms are developed for the case of coloured output noise.

The second thread, which considers the theory of errors-in-variables filtering for linear systems, extends the approach to deal with a class of bilinear systems, a frequently used subset of nonlinear systems. The application of errors-in-variables filtering for the purpose of system identification is also considered. This leads to the development of a prediction error method based on symmetric innovations, which resembles the joint output method. Both the offline and online implementation of this novel identification technique are investigated.

Acknowledgements

Firstly, I wish to sincerely thank and express my gratitude to Professor Keith J. Burnham for his continuing support throughout my PhD programme. I would also like to give special thanks to Dr. Benoit Vinsonneau for his valuable assistance, especially during the first part of my studies. In addition, I am very grateful to Tomasz Larkowski, Dr. Joe C. Whitehouse, Dr. David P. Goodall and Norbert Meyer, who supported and helped me via several interesting discussions.

To my family & Tara.

Contents

Nomenclature

Abbreviations

ARQ	Approximate Rayleigh quotient
BELS	Bias eliminating least squares
BCLS	Bias compensating least squares
BEIVKF	Errors-in-variables Kalman filter for bilinear systems
BKF	Bilinear Kalman filter
cf.	Confer (compare)
Ch.	Chapter
CG-RQ	Conjugate gradient subspace tracking algorithm (minimising Rayleigh quotient)
CM	Covariance-match (criterion)
dB	Decibel
EBCLS	Extended bias compensating least squares
EC	Extended bias compensating normal equations (criterion)
e.g.	Exempli gratia (for example)
EIV	Errors-in-variables
EIV-JEKF	Errors-in-variables Kalman filter for joint state and parameter estimation
EIV-RPEM	Recursive prediction error method for errors-in-variables systems
EIVKF	Errors-in-variables Kalman filter
EIVKF-JO	Errors-in-variables Kalman filter used within the joint output method
EKF	Extended Kalman filter
EM	Extended model (criterion)
flops	Floating point operations
FRFS	Fast recursive Frisch scheme
FSCON	Frisch scheme for coloured output noise
Frisch-CM	Frisch scheme using the CM model selection criterion
Frisch-EM	Frisch scheme using the EM model selection criterion
Frisch-EC	Frisch scheme using the EC model selection criterion
Frisch-YW	Frisch scheme using the YW model selection criterion
i.e.	Id est (that is)
IV	Instrumental variable
JEKF	Extended Kalman filter for joint state and parameter estimation
KF	Kalman filter
LS	Least squares
LTI	Linear time-invariant
LTV	Linear time-varying
MIMO	Multiple-input multiple-output
MISO	Multiple-input single-output
MSE	Mean square error

NLS	Nonlinear least squares
p.	Page
PEM	Prediction error method
PEM-SYM	Prediction error method based on symmetric innovations
RAFS	Repeatedly applied Frisch scheme algorithm
REBC	Recursive extended bias compensating algorithm
RBP	Recursive bilinear parametrisation algorithm
RBCLS	Recursive bias compensating least squares
RFS	Recursive Frisch scheme
RFSCON	Recursive Frisch scheme for coloured output noise
RIV	Recursive instrumental variable algorithm
RLS	Recursive least squares algorithm
RPEM	Recursive prediction error method
RPEM-SYM	Recursive prediction error method based on symmetric innovations
RVP	Recursive variable projection algorithm
SISO	Single-input single-output
SNR	Signal-to-noise ratio
YW	Yule-Walker (criterion)
YW-lin	Algorithm which minimises an approximate YW criterion
YW-lin-fast	YW-lin algorithm with reduced complexity

Notation

Latin variables

a	Coefficient vector of $A(q^{-1})$-polynomial
$\hat{a}_k$	Estimate after k samples
a_{LS}	Least squares solution of a
$A(q^{-1})$	Polynomial
$\hat{A}_k$	Schur complement
A_k, B_k, C_k, D_k, G_k, N_k	System matrices (non-errors-in-variables state space)
$\mathcal{A}_k$, $\mathcal{B}_k$, $\mathcal{C}_k$, $\mathcal{D}_k$, $\mathcal{G}_k$	System matrices (errors-in-variables state space)
$\mathcal{A}_0$, $\mathcal{B}_0$, $\mathcal{C}_0$, $\mathcal{D}_0$, $\mathcal{G}$	'True' system matrices (errors-in-variables state space)
$\mathcal{A}(\theta)$, $\mathcal{B}(\theta)$, $\mathcal{C}(\theta)$, $\mathcal{D}(\theta)$	Model matrices (general)
$\mathcal{A}_k^a$	Actual input dependent system matrix
$\mathcal{A}_k^d$	Design value for system matrix
$\hat{\mathcal{A}}_k$	Estimate of system matrix
$\mathscr{A}(\theta)$, $\mathscr{B}(\theta)$, $\mathscr{C}(\theta)$	System matrices (reformulated errors-in-variables-system)
b	Coefficient vector of $B(q^{-1})$-polynomial
$\hat{b}_k$	Estimate after k samples
b_{LS}	Least squares solution of b
$B(q^{-1})$	Polynomial
$B_k(\alpha_k)$	Schur complement (coloured output noise case)
$C(q^{-1})$	Polynomial
d_k	Noise term to model parameter variations
d_1^k, d_2^k, d_3^k, d_4^k, d_5^k	Auxiliary terms

$D(q^{-1})$	Polynomial
$D_{\mathcal{M}}$	Set of values over which θ ranges in a model structure
e_k	Output noise (in state space representation)
f	Function
$f(\sigma)$	Auxiliary vector
f_k	White noise
$F(\sigma)$	Auxiliary matrix
F_k	Jacobian
$\bar{F}_k$	Jacobian
F_1^k, F_2^k, F_3^k	Measures for Frisch-character
$\bar{F}$	Averaged Frisch-character
$F(\theta, x, u)$	Jacobian
g	Function
$g(\theta)$	Auxiliary vector
$G(\theta)$	Auxiliary matrix
G_k	Auxiliary matrix
h	Function
h_k	Auxiliary vector
H_k	Auxiliary matrix
H_k	Jacobian
$\bar{h}_k$	Auxiliary vector
$\bar{H}_k$	Auxiliary matrix
$H(\theta, x, u)$	Jacobian
H_k	Jacobian (depending on $\hat{\theta}_k$)
i	Discrete-time index
J	Auxiliary matrix
J^*	Auxiliary matrix (obtained by deleting the last row in J)
$J(\sigma_{\tilde{u}})$	Jacobian (of residual $r_k(\theta)$ with respect to $\sigma_{\tilde{u}}$)
$J(\vartheta)$	Jacobian (of residual $r_k(\theta, \sigma)$ with respect to ϑ)
$J_k^{(\sigma_{\tilde{u}})}$	Approximate Jacobian (of residual $r_k(\theta)$ with respect to $\sigma_{\tilde{u}}$)
$J(\theta, S^{-1}, \varepsilon)$	Jacobian
k	Number of samples, discrete-time index
K_k, $K_k(\theta)$	Kalman gain
$\mathscr{K}_k^{(i)}$	Derivative of $K_k(\theta)$ with respect to θ_i
l	Discrete-time index
L_k	Gain
$L_\theta(\vartheta)$	Linearised θ-equation of Frisch scheme
$L_{\sigma_{\tilde{y}}}(\vartheta)$	Linearised $\lambda_{\min}$-equation of Frisch scheme
$L_\theta^{k'}$	Recursively computed derivative of $L_\theta(\vartheta)$
m	Model order
M_u, M_y	Filter performance indices for input and output, respectively

MSE_k	Mean square error
n_a	Order of $A(q^{-1})$-polynomial
n_b	Order of $B(q^{-1})$-polynomial
n_c	Order of $C(q^{-1})$-polynomial
n_d	Order of $D(q^{-1})$-polynomial
n_θ	Dimension of parameter vector $n_\theta = n_a + n_b$
n_x	Dimension of state vector
n_z	Dimension of z_k (either system output or instrument vector)
P_k	(Scaled) error covariance matrix (recursive least squares)
$P_{k\|k-1}$ $P_{k\|k-1}(\theta)$	Error covariance matrix (Kalman filter)
P_0	Initial error covariance matrix (Kalman filter)
P_{1_k}, P_{2_k}, P_{3_k}	Partitioned covariance matrices
P_u^k	Error covariance matrix of filtered input
P_y^k	Error covariance matrix of filtered output
$\mathscr{P}_k^{(i)}$	Derivative of $P_{k\|k-1}(\theta)$ with respect to θ_i
P_{1_k}, P_{2_k}, P_{3_k}	Covariance matrices of Kalman filter for joint state and parameter estimation
$r_{cd}(\tau)$	Covariance (scalar)
$r_c(\tau)$	Covariance (scalar)
$r(\vartheta), r(\theta, \sigma)$	Nonlinear least squares residual
$r_k(\theta)$	Residual of Yule-Walker cost function
$\hat{r}_k$	Residual of conjugate gradient method
$R(x_k)$	Rayleigh quotient
R_k	Approximate Hessian
$s(\theta)$	Auxiliary vector
$S(\theta)$	Auxiliary matrix
S_k, $S_k(\theta)$	Innovations covariance matrix
$\mathscr{S}_k^{(i)}$	Derivative of $S_k(\theta)$ with respect to θ_i
$T(\theta)$, T_k	Auxiliary matrix
u_k	Measured (noise corrupted) input
u_{0_k}	Noise-free system input
$\hat{u}_{0_k}$	Filtered system input
$u_{0_k}^*$	Intermediate estimate of u_{0_k}
$\tilde{u}_k$	Input measurement noise
v_k	Process noise
v_k, $\bar{v}_k$, v_k^*, v_k'	Reformulated process noise
V_k	Yule-Walker cost function
$V_k^{(\theta)}$	Approximate derivative of V_k with respect to θ
$V_k^{(\sigma_{\tilde{u}})}$	Approximate derivative of V_k with respect to $\sigma_{\tilde{u}}$
V_{EC}	Cost function (model selection criterion)
$V(\theta)$	Cost function (prediction error method)
$\breve{V}(\theta)$	Cost function corresponding to modified predictor $\breve{y}_k(\theta)$
$V_\epsilon(\theta)$	Cost function corresponding to symmetric innovation $\epsilon_k(\theta)$

V_k^{lin}	Yule-Walker cost function using linearised Frisch scheme equations
V_1^k	Cost function for the determination of $\sigma_{\tilde{u}}$ (coloured output noise case)
$V_1^{k'}$, $V_1^{k''}$	First and second order derivative of V_1^k
V_2^k	Cost function for the determination of α_k (coloured output noise case)
$V_2^{k'}$, $V_2^{k''}$	First and second order derivative of V_2^k
V_k', V_k''	First and second order derivative of V_k
V_{LS}	Asymptotic least squares criterion
w_k	Process noise
W_k, $W_k(\theta)$	Jacobian of $\hat{x}_{k\|k-1}(\theta)$
$\breve{W}_k$, $\breve{W}_k(\theta)$	Jacobian of $\hat{x}_{k\|k-1}(\theta)$
x_k	State vector (filtering context)
x_k	Eigenvector corresponding to $\hat{A}_k$ (Frisch scheme context)
$\bar{x}_k$	Eigenvector x_k scaled to unity length
$\bar{x}_0$	Mean of initial state
$\hat{x}_{k\|k-1}$	State estimate
$\tilde{x}_k$	State estimation error
y_k	Measured (noise corrupted) output
y_{0_k}	Noise-free system output
$\hat{y}_{0_k}$	Filtered system output
$\tilde{y}_k$	Output measurement noise
$\breve{y}_k$, $\breve{y}_k(\theta)$	Modified predictor
z_k	Output of state space system (filtering context)
z_k	Instrument vector (extended bias compensating least squares context)
$z_{1_k}, z_{2_k}, z_{3_k}$	Instrument vectors
Z^k	Input/output data from time step 1 to time step k

Greek variables

α_k	Scaling factor
$\hat{\alpha}_k$	Scalar
β_i^k	Weighting of ith data at time instance k
γ_k	Gain sequence or step size
δ_k	Instrument vector
δ_k	Extended instrument vector comprising delayed inputs
δ_k^*	Obtained by deleting last entry in δ_k
ϵ_k	Residual
ε_k	Innovation
ϵ_k, $\epsilon_k(\theta)$	Symmetric innovation
$\breve{\varepsilon}_k$, $\breve{\varepsilon}_k(\theta)$	Innovation corresponding to modified predictor $\breve{y}_k(\theta)$
ζ_k	Instrument vector comprising delayed inputs

θ	Parameter vector
θ_0	'True' parameter vector describing the system
$\hat{\theta}_k$	Estimate after k samples
$\hat{\theta}_k^{\mathrm{LS}}$	Least squares estimate
$\hat{\theta}_k^{\mathrm{IV}}$	Instrumental variable estimate
$\bar{\theta}$	Extended parameter vector
$\theta_k^{(\tilde{y})}$	Approximate derivative of $\hat{\theta}_k$ with respect to $\hat{\sigma}_{\tilde{y}}^k$
$\theta_k^{(\tilde{u})}$	Approximate derivative of $\hat{\theta}_k$ with respect to $\hat{\sigma}_{\tilde{u}}^k$
$\hat{\theta}_{k+\frac{1}{2}}$	Intermediate estimate of θ
Θ	Augmented parameter vector (coloured output noise case)
ϑ	Augmented parameter vector (white output noise case)
ϑ_*	Point of linearisation
$\iota(\vartheta)$	Auxiliary term
$\bar{\iota}(\vartheta)$	Auxiliary term
$\kappa(\vartheta)$	Auxiliary term
λ_k	Forgetting factor
Λ	Weighting matrix
$\hat{\mu}_k$	Step size (conjugate gradient)
ξ_{vc}	Covariance vector
$\hat{\xi}_{vc}^k$	Estimated covariance vector
$\rho_{\tilde{y}}$	Vector of auto-correlation elements of $\tilde{y}_k$
σ	Vector comprising noise variances
σ_e	Variance of reformulated output noise
Σ_ε^k	Innovation covariance matrix
σ_w	Variance of process noise
σ_{u_0}, σ_{y_0}	Variance of noise-free input and noise-free output, respectively
σ_u, σ_y	Variance of measured input and measured output, respectively
$\sigma_{\tilde{u}}$	Input measurement noise variance
$\sigma_{\tilde{u}}^{\max}$	Maximal admissible value for $\hat{\sigma}_{\tilde{u}}^k$
$\sigma_{\tilde{u}\tilde{y}}$	Covariance of input and output measurement noise variance
$\sigma_{\tilde{y}}$	Output measurement noise variance
$\sigma_{\tilde{y}}^{\max}$	Maximal admissible value for $\hat{\sigma}_{\tilde{y}}^k$
$\hat{\varsigma}_k$	Input measurement noise variance (using $\lambda_{\min}$-equation)
$\sigma_{\tilde{y},k}^{(\tilde{u})}$	Approximate derivative of $\hat{\sigma}_{\tilde{y}}^k$ with respect to $\hat{\sigma}_{\tilde{u}}^k$
Σ_{vw}	Covariance matrix
Σ_v	Covariance matrix
$\hat{\Sigma}_{vw}^k$	Covariance matrix estimate
$\bar{\Sigma}_{vw}$	Covariance matrix estimate using weighted arithmetic mean
$\Sigma_{\tilde{\varphi}}(\sigma)$	Noise compensation matrix
Σ_v^k, $\Sigma_{\bar{v}}^k$, $\Sigma_{v^*}^k$, $\Sigma_{v'}^k$	Auto-covariance matrix of reformulated process noise
Σ_{ve}^k, $\Sigma_{\bar{v}e}^k$, $\Sigma_{v^*e}^k$, $\Sigma_{v'e}^k$	Cross-covariance matrix of reformulated process noise and reformulated output noise
Σ_x^k	Auto-covariance matrix of the state vector

$\Sigma_{x\tilde{x}}^k$	Cross-covariance matrix of the state vector and state estimation error
φ_k	Noisy regression vector
φ_{0_k}	Noise-free regression vector
$\tilde{\varphi}_k$	Noise part of regression vector
$\bar{\varphi}_k$	Extended noisy regression vector
$\bar{\varphi}_{0_k}$	Extended noise-free regression vector
$\tilde{\bar{\varphi}}_k$	Extended noise part of regression vector
$\hat{\psi}_k$	Conjugate gradient update direction
$\hat{\psi}_k$	Gauss-Newton update direction corresponding to ϑ
$\hat{\psi}_k^\theta$	Gauss-Newton update direction corresponding to θ
$\hat{\psi}_k^\theta$	Gauss-Newton update direction corresponding to σ
ψ_k, $\psi_k(\theta)$	Gradient of predictor

Operators and symbols

$\triangleq$	Equal by definition
$\in$	Element in
$\|\cdot\|$	Norm of matrix
$\otimes$	Kronecker product
$\times$	'Times', or Cartesian product, depending on the context
$\mathbb{R}^{m\times n}$	$m \times n$ dimensional space of real numbers
$\mathcal{R}(A)$	Range of matrix A
$\arg\min f(x)$	Value of x that minimises $f(x)$
I_n	Identity matrix of size n
$0_{n\times m}$	Zero matrix of dimension $n \times m$
$\det(M)$	Determinant of matrix M
$\mathrm{diag}(M)$	Diagonal elements of matrix M
$\mathrm{rank}(M)$	Rank of matrix M
$E[\cdot]$	Expected value operator
M^T	Transpose of matrix M
M^{-1}	Inverse of matrix M
$M^\dagger$	Moore-Penrose pseudo inverse of matrix M
$N(M)$	Nullspace of matrix M
$O(\cdot)$	'Big-O-notation'
q^{-1}	Backward shift operator
δ_{kl}	Kronecker delta function (zero unless $k = l$)
$\lambda_{\min}(M)$	Minimum eigenvalue of matrix M
$\rho(M)$	Spectral radius of matrix M

List of Algorithms

Chapter 1

Introduction and outline of approach

Contents

1.1 Introduction

A system is a real-world entity, whose behaviour is of interest for various reasons such as improved control system design, condition monitoring or fault detection. For instance a marine vessel can be considered as being a system for which the knowledge of its behaviour is required in order to design an appropriate dynamic positioning control strategy. Building mathematical models of real-world systems via measured data is a common problem not only in engineering, but also in natural sciences, social sciences, finance and the manufacturing industries. The process of building mathematical models based on observed data from the system is called *system identification* (Ljung 1999). For a system, it is usual to distinguish between the input signals, which drive the system

and can eventually be manipulated, and the output signals, which are considered to be the observed values of interest. A frequent assumption in system identification is that the input variables are known exactly. However, it might often be the case that not only the outputs, but also the inputs are subject to additive measurement noise. Using again the marine vessel as an exemplary system, the input to the system could be the force provided by propellers, which move the vessel through the water, whilst the position could be regarded as a resulting output. Since the exact force which is driving the vessel is generally unknown, but can be deduced by measuring the rotational speed of the propellers, the input signal will be affected by uncertainties. In such cases, 'classical' identification methods may fail to yield consistent parameter estimates, i.e. the estimated parameters do not converge to the 'true' values with an increasing number of measurements. Especially when the parameters reflect physical meaning, and depending on the particular application, a systematic estimation error (bias) may lead to significant problems. This prompts the need for a so-called *errors-in-variables* (EIV) approach, where the input noise is taken into account to obtain unbiased estimates of the model parameters.

An overview of recent developments for the identification of dynamical EIV systems is given in (Söderström 2007*b*, Söderström, Soverini & Mahata 2002, Söderström 1981). Identifying EIV systems is strongly related to the total least squares (TLS) problem, also known as orthogonal regression, which is extensively treated in (Van Huffel & Vandewalle 1991, Van Huffel 1997, Van Huffel & Lemmerling 2002). One particular EIV identification technique, which has received significant attention in recent years, is the so-called Frisch scheme, which dates back to (Frisch 1934). Whilst originally developed to deal with an algebraic regression problem, the Frisch scheme has been extended towards the dynamic case in (Beghelli, Guidorzi & Soverini 1990) with further analysis and extensions reported in (Guidorzi & Pierantoni 1995, Diversi, Guidorzi & Soverini 2006, Söderström 2007*a*, Hong, Söderström, Soverini & Diversi 2007, Söderström 2008). Another EIV system identification approach is the extended bias compensating least squares (EBCLS) technique, which has been developed in (Ekman 2005). Similar to the Frisch scheme, the EBCLS approach is based on the bias correction principle (Sagara & Wada 1977, Stoica & Söderström 1982, Zheng & Feng 1989), which exploits the fact that the asymptotic bias of the least squares (LS) solution can be removed if the variances of the input and output measurement noise sequences are known, or estimated. The relation between the Frisch scheme and the EBCLS approach has recently been analysed in (Hong & Söderström 2008).

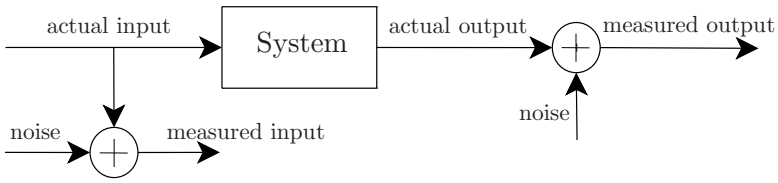
In many applications it is required to obtain, or continually update, a model while the process which is generating the data is operating. In this case, the parameter estimation procedure is required to be carried out in an online manner rather than making use of a previously collected batch of data, the latter being termed offline identification. Online system identification can be achieved by making use

of *recursive identification techniques*, which are thoroughly discussed in (Ljung & Söderström 1983, Young 1984, Kushner & Yin 2003). Recursive identification techniques continually update a currently obtained model at each successive sample instance, as new data becomes available. Motivation for recursive system identification stems from adaptive control, fault detection and diagnostics, adaptive signal processing, adaptive filtering and/or its application for offline identification. Note that the topic of recursive identification is also known as real-time identification or sequential estimation within the literature.

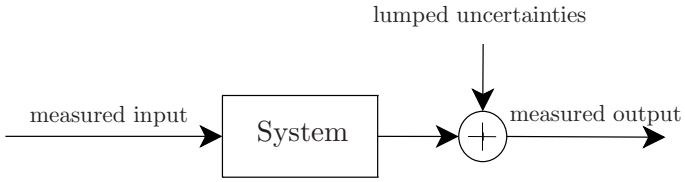
Closely related to the recursive system identification problem is *filtering*, a term which is used here to describe the estimation of signals from noise corrupted measurements. Whilst the fundamental principle of filtering is probably as old as mankind itself, digital filter design employing statistical ideas dates back to the work of Kolmogorov and Wiener (Kolmogorov 1941, Wiener 1964), where stationary random processes are considered. The more general nonstationary case is dealt with by the famous Kalman filter (KF) (Kalman 1960), which may be regarded as one of the most widely applied tools in modern engineering and applied mathematics. A thorough treatment of Kalman filter theory is given by (Jazwinski 1970, Anderson & Moore 1979, Kailath & Sayed 2000), whilst a historical survey can be found in (Sorenson 1970). When applied to estimate the states and the outputs of linear systems driven by a deterministic input signal (and possibly in the presence of process noise), one of the assumptions within the KF is that the input signal is known exactly. However, as in the case of system identification, this assumption may not always be fulfilled. This potential shortcoming has stimulated researchers to consider Kalman filtering within an extended noise environment, which has led to the recent development of an errors-in-variables Kalman filter (EIVKF) (Guidorzi, Diversi & Soverini 2003, Markovsky & De Moor 2005, Diversi, Guidorzi & Soverini 2005), which allows, under certain assumptions, an estimate of the system states, the noise-free output signals as well as the noise-free input signals to be obtained.

Another common assumption in system identification as well as filtering, due mainly to the wealth of well developed theory, is the linearity of the underlying system or process leading to the relative simplicity of the required mathematics as well as the resulting algorithms. Whilst most (if not all) real-world processes are of a nonlinear nature, it is possible and sufficient in a large number of situations to approximate the nonlinear process using a linear model structure. One possibility to retain the desirable features of the well structured linear case, but simultaneously allowing a successful extension to encompass and approximate a large number of nonlinear systems over a wider range of operation, is to utilise so-called *bilinear models* (Pearson 1999); a simple class of nonlinear models with a far reaching applicability.

The first of the two main threads within the thesis is that of developing recursive algorithms for the identification of EIV systems based on the (offline) Frisch scheme



(a) Errors-in-variables setup.



(b) Classical identification setup.

Figure 1.1: Errors-in-variables setup versus classical setup.

as well as the EBCLS approach. A further strand in this area is that of developing fast algorithms. Within the second thread, the theory of linear EIV filtering is firstly extended towards bilinear system representations and secondly applied to derive EIV identification techniques, which are considered to be novel.

The remainder of this introductory chapter is outlined as follows. Section 1.2 describes the motivation for the above developments within this thesis and Section 1.3 formulates the problems which are addressed. The methodology as well as a brief outline for each of the forthcoming chapters is given in Section 1.4, whilst Section 1.5 itemises the contributions of the author.

1.2 Motivation

1.2.1 Why errors-in-variables?

A frequent assumption within the system identification literature (e.g. Ljung 1999) is that the input of the system is known exactly. However, in many practical situations the actual input, which is driving the system, is unknown and must be measured or estimated, hence the measured variable will be corrupted by noise. Such a setup is depicted in Figure 1.1(a). Classical prediction error approaches follow a pragmatic approach and regard the measured input as the actual input whilst lumping all uncertainties to the output of the system, as illustrated in the classical setup in Figure 1.1(b). This might be a reasonable assumption if the input measurement uncertainties are negligible and/or if the prediction of future output values is the major objective of the identification task. If, in contrast, the aim is to obtain a better understanding of the underlying relations between noise-free inputs and noise-free outputs, i.e. if the parameters themselves are meaningful and of interest, then it is often beneficial to explicitly account for the input measurement uncertainties. This leads naturally to the problem of considering the effects of errors on the output as well as the errors

on the input variable. In general, the difficulty of the identification problem increases when considering an EIV approach. In addition, the resulting algorithms are often more complex; conceptually and/or computationally. If the input disturbance is not taken into account, the resulting parameter estimates usually exhibit an asymptotic bias, i.e. no matter how many samples are utilised for the identification task, the estimates obtained will not converge to the 'true' parameter vector. Obtaining unbiased estimates of the parameter vector, if the values of the latter are meaningful (e.g. they correspond to physical quantities), is considered to be the 'classical' motivation for EIV identification (Söderström 2007*b*). Another motivation for adopting an EIV approach is that the system can be considered to be symmetric, i.e. there is, in principle, no need to distinguish between inputs and outputs since both are treated equally. This is also related to the behavioural framework (Polderman & Willems 1998), an alternative approach to dynamic system modelling.

It is worth mentioning that in the case of static systems, the EIV concept relates to other well-known topics such as latent variables and factor models (cf. references within Söderström 2007*b*). The applicability of EIV modelling techniques is spread over many areas including econometrics, engineering and finance to name only a few.

1.2.2 Why recursive identification?

Recursive identification techniques aim to update a model while the system which is generating the data is operating in real time. One area of application is that of fault detection and diagnostics, where the aim is to detect incipient faults in hardware and/or software before the conditions deteriorate seriously. Naturally, faults may occur when the process is running and it is therefore required to detect these as early as possible, in order to prevent major damage. Changes in the system parametrisation, either abruptly or subtly, can often indicate such fault conditions within the system, which clearly prompts the need for online system identification. Indeed, the fault detection scenario outlined here motivates an immediate need for both online and EIV identification, since a fault is often indicated by changes in the physical properties of a system. Using an appropriately parameterised model, this leads naturally to a requirement for model parameters to be estimated recursively as well as consistently. To the best knowledge of the author, the recursive EIV identification problem has, however, received rather limited attention within the literature (see e.g. Chen 2007, Ding, Chen & Qiu 2006, Chen & Yang 2005, Zheng & Feng 1989). Another motivation for recursive EIV identification stems from that of adaptive control system design. Assume that a pole placement controller is to be updated based on an identified model of the system. If the estimated poles of the system are biased (e.g. due to the presence of input disturbances) the closed-loop poles will be 'misplaced' and the control performance may differ significantly from the specified design. Hence, unbiased online estimates are required, which again clearly prompts the need for recursive EIV identification techniques. In

general, the use of recursive identification schemes can also give an additional insight into the functionality of the algorithms, since the trajectories of the estimates reveal the 'history' of the estimates. The trajectories can, for instance, indicate problems due to identifiability or ill-conditioning, which the engineer can detect by inspection. Last but not least, recursive algorithms are able to track changes in the operating conditions of a system, which might result in time varying parameters of the identified model, hence allowing a wider range of applicability. Note that tracking parameter changes, which might relate to changes in the operating conditions of a system, requires some form of adaptation, and it is the recursive structure of the algorithms, which provides the natural environment to realise such adaptivity.

Computational complexity of recursive algorithms

Another important issue within this thesis deals with the reduction of computational complexity of the recursive algorithms. On the one hand, persistently increasing capacities of modern digital computers allow for the implementation and realisation of increasingly complex algorithms. For instance, whilst being unthinkable only a few years ago, computationally demanding methods, such as the particle filter approach for nonlinear filtering (Arulampalam, Maskella, Gordon & Clapp 2002), find many practical applications nowadays. On the other hand, arguing that such an explosion in computation power eclipses the need for the development of computationally parsimonious algorithms, is a sophism: many companies use the cheapest hardware possible, in order to maximise their profits. In addition they are restricted to the use of established technologies, since the introduction of new hardware often accompanies a lengthy verification process, which implies additional costs, hence an option often avoided. As a consequence, today's industrial engineers are still confronted with severe restrictions concerning the computational complexity of their algorithms. Therefore, the development of computationally economic algorithms is naturally motivated by practical needs of industry. Another justification is given by the fact that the increase in computational power allows a reduction of the sampling interval and potentially permits the control of higher bandwidth systems. However, the reduced sampling interval will inevitably continue to limit the available operating time for online algorithms (which usually perform their entire operations in between samples). Finally, note that due to the increased complexity of some EIV identification algorithms (with respect to non-EIV approaches), a reduction in computation time might be a desired feature for recursive implementations. Whilst an upper limit of the computation time for an online algorithm will ultimately depend on the targeted application, the above discussion indicates the importance of minimising the computation time of recursive EIV identification algorithms, which is also addressed within this thesis.

1.2.3 Why errors-in-variables filtering?

Filtering aims to reduce the effect of additive noise on signals, hence is a subject of paramount importance not solely in branches of science and engineering, but also in many other areas. When both outputs and inputs are corrupted by uncertainties, it is naturally of interest to filter both quantities, i.e. to remove the noise not only from the output signal, but also from the input signal. As for filtering in general, a further motivation for EIV filtering is its use for system identification. It is often possible to cross-couple filtering and identification, i.e. to use the filtered signals to improve the parameter estimates, which in turn can be utilised to improve the filter performance and so on. This can lead to online as well as offline identification techniques, which motivates the need for filtering in general, and here, in particular, EIV filtering.

1.3 Statement of the problem

The thesis follows two fundamental threads of research: recursive Frisch scheme identification and EIV filtering.

1.3.1 Recursive Frisch scheme identification

The first thread deals with the development of recursive system identification algorithms based on the offline Frisch scheme for errors-in-variables system identification. To the best knowledge of the author, this problem has not to date been considered within the literature. Within this framework, two cases are considered:

1. Both the input measurement noise sequence and the output measurement noise sequence are white, i.e. uncorrelated in time.

2. The input measurement noise sequence is white, whereas the output measurement noise sequence is coloured, i.e. correlated in time.

Hence, the objective is the development of recursive EIV identification techniques which exactly, or at least approximately, update the estimates using the Frisch scheme. The development follows two approaches:

(a) Recursive algorithm design based on the offline Frisch scheme.

(b) Recursive algorithm design using the EBCLS approach.

In addition, attention is given to the development of computationally parsimonious algorithms, in order to allow a wide range of applications.

1.3.2 Errors-in-variables filtering

The second thread within this thesis deals with the EIV filtering problem. Whilst this has been solved for the linear case (see e.g. Guidorzi et al. 2003), it has not been

considered for bilinear systems within the literature. Therefore, the first objective within this framework is the development of bilinear EIV filter algorithms. The second objective concerns the application of EIV filtering for the purpose of offline as well as online system identification, hence linking the two individual threads within the thesis.

1.4 Outline of approach

1.4.1 Methodology

In mathematics, natural science and engineering, novel developments usually build on well established foundations of theory and methodology. The underlying approach within this thesis is to develop novel techniques and algorithms which build on well-known concepts of EIV system identification and EIV filtering. This is achieved by either modifying and extending existing techniques towards more general settings, focusing on different aspects and/or by combining existing theory and methods towards novel settings. Thereby, importance is attached to providing detailed listings of the developed algorithms in a form which allows a straightforward implementation in commonly used software packages, such as Matlab. Where possible and appropriate, mathematical development is substantiated and demonstrated in numerical simulations, where care has been taken in order to ensure reproducibility of the latter.

Commonly used notation is introduced in Chapter 2 whilst additional chapter-specific notation and abbreviations are introduced when required. A list of all abbreviations used is given in a global nomenclature section, which also provides an extensive list of the mathematical notation which is used in the overall thesis. In addition, chapter-specific notation is summarised within a nomenclature section at the beginning of each chapter. At the commencement of Chapters 3-7 a list of preliminary reading from specific underpinning sections in the thesis is given. Note that the reader might only require parts of the review Chapter 2 in order to follow the developments in the individual chapters. A brief outline for the subsequent chapters is given in the following subsection.

1.4.2 Chapter outlines

Chapter 2: The purpose of this chapter is to review well known techniques and principles of mathematics, system identification and Kalman filtering, which provide the foundation for the novel developments in the subsequent chapters. In particular, it reviews the offline identification schemes which are extended and modified for the purpose of recursive estimation, which is the topic of Chapters 3-5. Moreover, it reviews errors-in-variables filtering, forming the basis for Chapters 6-7.

Chapter 3: Based on the offline algorithm, this chapter derives recursive expressions for the Frisch scheme equations using the Yule-Walker model selection criterion.

The computational complexity of the recursive algorithms is analysed in detail and bottlenecks are identified. Potential shortcomings of the algorithms are discussed together with possibilities for further improvements.

Chapter 4: This chapter is divided into two parts: The first part builds on the development of Chapter 3 and devises, by introducing additional approximations, fast recursive Frisch scheme algorithms whose computational complexity is reduced from cubic to second order. The second part considers the coloured output noise case and develops two recursive algorithms within this framework.

Chapter 5: Recursive Frisch scheme algorithms are developed within this chapter, by interpreting the Frisch scheme using the Yule-Walker model selection criterion as an extended bias compensating least squares problem. Bilinear parametrisation approaches as well as variable projection algorithms are considered. In addition, an extensive simulation study provides a comparison of the developed algorithms with those developed in Chapters 3 and 4.

Chapter 6: This chapter considers the problem of EIV filtering, i.e. the estimation of the noise-free input and noise-free output signals, for a class of bilinear time-invariant single-input single-output state space systems. It is shown that the optimal filter, in a minimum variance sense, is infeasible, since its design depends on the knowledge of the noise-free input. Consequently, four suboptimal approaches are developed and compared via simulation.

Chapter 7: The use of EIV filtering for the purpose of parameter estimation is investigated in this chapter. Initially, the extended Kalman filter for joint state and parameter estimation (JEKF) is derived for EIV state space systems. Then, a recursive prediction error method (RPEM) when applied to an EIV state space system is studied. Use of the filtered inputs and outputs leads to a modified JEKF and RPEM design which is able to reduce the asymptotic bias in the parameter estimates. A novel EIV identification method based on symmetric innovations is derived, which resembles the well-known joint output method for EIV system identification. Offline and online implementations are investigated.

A 'road map' for the outline of the thesis, which indicates the dependencies between the various chapters and provides a guideline for the reader, is illustrated in Figure 1.2.

1.5 Contributions

The contributions of the author are listed here in descending order with respect to their considered significance.

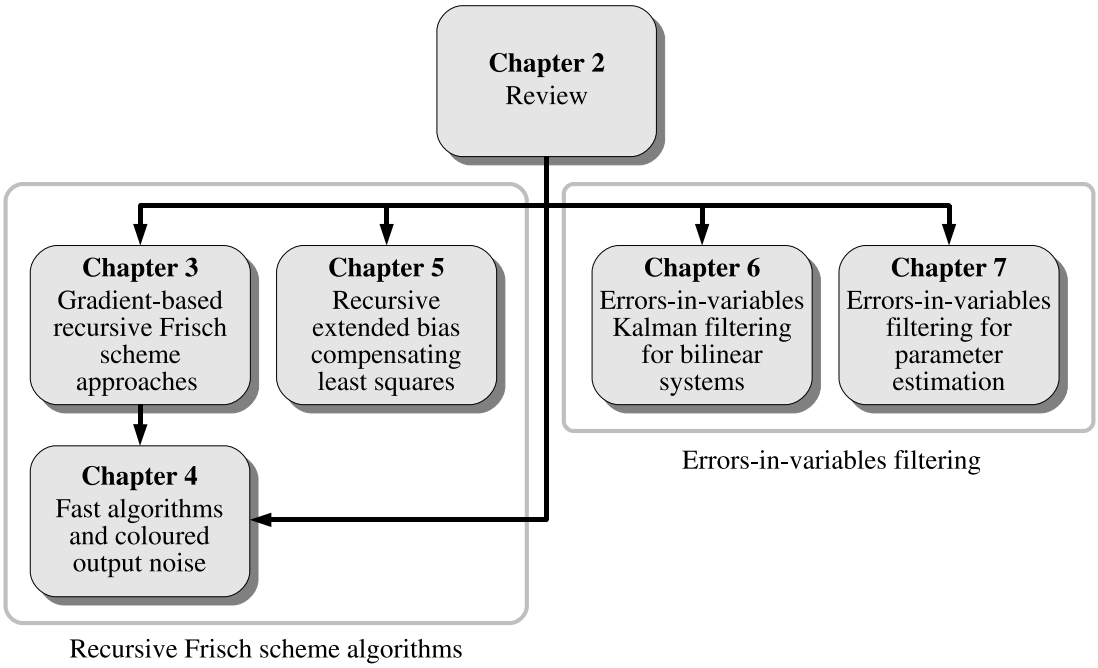


Figure 1.2: Road map outline of the thesis.

1. *Development of recursive algorithms based on the offline Frisch scheme:* Two algorithms have been developed which allow an (approximate) update of the parameter estimates obtained by the offline Frisch scheme. A discussion concerning shortcomings has been provided together with suggestions for potential improvements. Parts of this work have also been published in:

   [1] Linden, J. G., Meyer, N., Vinsonneau, B. & Burnham, K. J. (2006), Recursive Frisch scheme identification incorporating adaptivity, *in* 'Proc. DVD-ROM 21st IAR & ACD Workshop', Nancy, France.

   [2] Linden, J. G. & Burnham, K. J. (2008), Some aspects on recursive Frisch scheme identification, *in* 'Computer Systems Engineering, Proc. 6th & 7th Polish British Workshop', pp. 176–187.

   [3] Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2008), Gradient-based approaches for recursive Frisch scheme identification, *in* 'Preprints of the 17th IFAC World Congress', Seoul, Korea, pp. 1390–1395.

   [4] Linden, J. G., Larkowski, T. & Burnham, K. J. (2008), An improved recursive Frisch scheme identification algorithm, *in* 'Proc. 19th Int. Conf. on Systems Engineering', Las Vegas, USA, pp. 65–70.

2. *Development of a symmetric prediction error method for errors-in-variables identification:* Analogously to the joint output method, EIV Kalman filtering is utilised for the development of novel offline and online identification techniques for a class of EIV systems. Early approaches have been published in:

[5] Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2007), An investigation of extended Kalman filtering in the errors-in-variables framework - A joint state and parameter estimation approach, *in* 'Proc. Int. Conf. on Informatics in Control, Automation and Robotics', Angers, France, pp. 47–53.

3. *Development of fast recursive Frisch scheme algorithms:* Building on the proposed recursive algorithms based on the offline Frisch scheme, computationally less expensive procedures have been developed. This work is based on the development in:

   [6] Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2007), Fast algorithms for recursive Frisch scheme system identification, *in* 'Proc. CD-ROM 22nd IAR & ACD Workshop', Grenoble, France.

4. *Development of bilinear errors-in-variables Kalman filters:* The theory of EIV Kalman filtering has been extended to deal with a class of bilinear EIV systems. Four suboptimal Kalman filters have been proposed within this framework. Part of this work is based on:

   [7] Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2007), Errors-in-variables filtering approaches for bilinear systems, *in* A. Grzech, ed., 'Proc. of 16th Int. Conf. Systems Science', Vol. 1, Wroclaw, Poland, pp. 446–455.

5. *Development of recursive extended bias compensating least squares algorithms:* Four recursive algorithms have been proposed to estimate the parameters of an EIV system within the EBCLS framework. Some of the development has been published in:

   [8] Linden, J. G. & Burnham, K. J. (2008), Recursive Frisch scheme identification via variable projection, *in* 'Proc. CD-ROM 11th Mechatronics Forum Biennial International Conference', Limerick, Ireland.

6. *Development of recursive Frisch scheme algorithms in the case of coloured output noise:* Two algorithms have been developed which update the estimate of the Frisch scheme for the case where the output noise sequence is correlated in time. Part of this development has been published in:

   [9] Linden, J. G. & Burnham, K. J. (2008), A recursive Frisch scheme algorithm for coloured output noise, *in* 'Proc. Int. Conf. on Informatics in Control, Automation and Robotics', Madeira, Portugal, pp. 163–170.

Chapter 2

Review

Contents

Nomenclature

a	Coefficient vector of $A(q^{-1})$-polynomial
$\hat{a}_k$	Estimate after k samples
$A(q^{-1})$	Polynomial
A_k, B_k, C_k, D_k, G_k, N_k ..	System matrices (non-errors-in-variables state space)
$\mathcal{A}_k$, $\mathcal{B}_k$, $\mathcal{C}_k$, $\mathcal{D}_k$, $\mathcal{G}_k$	System matrices (errors-in-variables state space)
$\mathscr{A}(\theta), \mathscr{B}(\theta), \mathscr{C}(\theta)$	System matrices (reformulated errors-in-variables-system)

b	Coefficient vector of $B(q^{-1})$-polynomial
$\hat{b}_k$	Estimate after k samples
$B(q^{-1})$	Polynomial
$B_k(\alpha_k)$	Schur complement
$C(q^{-1})$	Polynomial
d_k	Noise term to model parameter variations
$D(q^{-1})$	Polynomial
$D_{\mathcal{M}}$	Set of values over which θ ranges in a model structure
e_k	Output noise (in state space representation)
f	Function
$f(\sigma)$	Auxiliary vector
f_k	White noise
$F(\sigma)$	Auxiliary matrix
F_k	Jacobian
$\bar{F}_k$	Jacobian
g	Function
$g(\theta)$	Auxiliary vector
$G(\theta)$	Auxiliary matrix
G_k	Auxiliary matrix
h	Function
h_k	Auxiliary vector
H_k	Auxiliary matrix
H_k	Jacobian
$\bar{h}_k$	Auxiliary vector
$\bar{H}_k$	Auxiliary matrix
J	Auxiliary matrix
i	Discrete time index
I_n	Identity matrix of size n
k	Number of samples, discrete time index
K_k	Kalman gain
l	Discrete time index
L_k	Gain
n_a	Order of $A(q^{-1})$-polynomial
n_b	Order of $B(q^{-1})$-polynomial
n_c	Order of $C(q^{-1})$-polynomial
n_d	Order of $D(q^{-1})$-polynomial
n_θ	Dimension of parameter vector $n_\theta = n_a + n_b$
n_x	Dimension of state vector
n_z	Dimension of z_k (either system output or instrument vector)
P_k	(Scaled) error covariance matrix (recursive least squares)
$P_{k\|k-1}$	Error covariance matrix (Kalman filter)
P_0	Initial error covariance matrix (Kalman filter)
P_{1_k}, P_{2_k}, P_{3_k}	Partitioned covariance matrices
P_u^k	Error covariance matrix of filtered input
P_y^k	Error covariance matrix of filtered output
q^{-1}	Backward shift operator
$r_{cd}(\tau)$	Covariance (scalar)
$r_c(\tau)$	Covariance (scalar)
$r(\vartheta), r(\theta, \sigma)$	Nonlinear least squares residual
S_k	Innovations covariance matrix

u_k	Measured (noise corrupted) input
u_{0_k}	Noise-free system input
$\hat{u}_{0_k}$	Filtered system input
$\tilde{u}_k$	Input measurement noise
v_k, w_k	Process noise
V_k	Yule-Walker cost function
V_{EC}	Cost function
V_k^{lin}	Yule-Walker cost function using linearised Frisch scheme equations
V_1^k	Cost function (coloured output noise case)
V_2^k	Cost function (coloured output noise case)
$V_\epsilon(\theta)$	Cost function
x_k	State vector
$\bar{x}_0$	Mean of initial state
$\hat{x}_{k\|k-1}$	State estimate
y_k	Measured(noise corrupted) output
y_{0_k}	Noise-free system output
$\hat{y}_{0_k}$	Filtered system output
$\tilde{y}_k$	Output measurement noise
z_k	Output of (non-errors-in-variables) state space system
$z_{1_k}, z_{2_k}, z_{3_k}$	Instrument vectors
Z^k	Input/output data up from 1 to time k
α_k	Scalar
$\hat{\alpha}_k$	Scalar
β_i^k	Weighting of ith data at time instance k
γ_k	Normalising gain
δ_k	Instrument vector
δ_{kl}	Kronecker delta function
ϵ_k	Residual
ε_k	Innovation
$\epsilon_k(\theta)$	Symmetric innovation
ζ_k	Instrument vector
θ	Parameter vector
θ_0	'True' parameter vector describing the system
$\hat{\theta}_k$	Estimate after k samples
$\hat{\theta}_k^{\mathrm{LS}}$	Least squares estimate
$\bar{\theta}$	Extended parameter vector
Θ	Augmented parameter vector in coloured output noise case
ϑ	Augmented parameter vector
λ_k	Forgetting factor
$\lambda_{\min}(\cdot)$	Minimum eigenvalue operator
ξ_{vc}	Covariance vector
$\hat{\xi}_{vc}^k$	Estimated covariance vector
$\rho_{\tilde{y}}$	Vector of auto-correlation elements of $\tilde{y}_k$
$\rho(A)$	Spectral radius of matrix A
σ	Vector comprising noise variances
$\sigma_{\tilde{u}}$	Input measurement noise variance
$\sigma_{\tilde{u}}^{\max}$	Maximal admissible value for $\hat{\sigma}_{\tilde{u}}^k$
$\sigma_{\tilde{y}}$	Output measurement noise variance
$\sigma_{\tilde{y}}^{\max}$	Maximal admissible value for $\hat{\sigma}_{\tilde{y}}^k$
$\hat{\varsigma}_k$	Input measurement noise variance (using $\lambda_{\min}$-equation)

Σ_{vw}	Covariance matrix
Σ_v	Covariance matrix
$\hat{\Sigma}_{vw}^k$	Covariance matrix estimate
$\bar{\Sigma}_{vw}$	Covariance matrix estimate using weighted arithmetic mean
$\Sigma_{\tilde{\varphi}}(\sigma)$	Noise compensation matrix
φ_k	Noisy regression vector
φ_{0_k}	Noise-free regression vector
$\tilde{\varphi}_k$	Noise part of regression vector
$\bar{\varphi}_k$	Extended noisy regression vector
$\bar{\varphi}_{0_k}$	Extended noise-free regression vector
$\tilde{\bar{\varphi}}_k$	Extended noise part of regression vector
$O(\cdot)$	'Big-O-notation' indicating order of complexity
$\det(M)$	Determinant of matrix M
$E[\cdot]$	Expected value operator
$M^\dagger$	Moore-Penrose pseudo inverse of M
$N(M)$	Nullspace of M

2.1 Introduction

This chapter aims to build the foundation for the forthcoming chapters. Firstly, Section 2.2 introduces some common notation which is utilised throughout the whole thesis. Section 2.3 introduces the concept of iterative methods for least squares, the variable projection principle as well as the concept of counting the number of floating point operations in order to measure the computational complexity of algorithms. These techniques find several applications in the forthcoming chapters. Section 2.4 reviews some offline errors-in-variables (EIV) identification techniques: Initially, the fundamental bias compensation principle, which forms the basis for most of the algorithms within this thesis, is reviewed. Next, attention is focused in some detail on the Frisch scheme. Within this context, identification techniques for white and coloured output noise are considered. The review of EIV identification techniques is concluded with the extended bias compensation least squares technique and the joint output method. Section 2.5 then departs from the system identification framework and focuses on filtering. Following a review of the standard Kalman filter applied to linear systems, the concept of EIV filtering is introduced. Subsequently, the Kalman filtering problem for bilinear systems is considered whilst the section concludes by reviewing the extended Kalman filter for joint state and parameter estimation.

2.2 Notational conventions

This section introduces the commonly used notation within this thesis. If not stated otherwise, a discrete-time linear time-invariant (LTI) single-input single-output (SISO)

EIV system is considered, which is described by

$$A(q^{-1})y_{0_k} = B(q^{-1})u_{0_k}, \tag{2.1}$$

where k is an integer valued time index and

$$A(q^{-1}) \triangleq 1 + a_1 q^{-1} + ... + a_{n_a} q^{-n_a}, \tag{2.2a}$$
$$B(q^{-1}) \triangleq b_1 q^{-1} + ... + b_{n_b} q^{-n_b}, \tag{2.2b}$$

are polynomials in the backward shift operator q^{-1}, which is defined such that $q^{-i}x_k = x_{k-i}$. The noise-free input u_{0_k} and output y_{0_k} are unknown and only the measurements

$$u_k = u_{0_k} + \tilde{u}_k, \tag{2.3a}$$
$$y_k = y_{0_k} + \tilde{y}_k \tag{2.3b}$$

are available, where $\tilde{u}_k$ and $\tilde{y}_k$ denote the input and output measurement noise, respectively. Such a setup is depicted in Figure 2.1.

The parameter vector and an extended version is defined as

$$\theta \triangleq \begin{bmatrix} a^T & b^T \end{bmatrix}^T = \begin{bmatrix} a_1 & ... & a_{n_a} & b_1 & ... & b_{n_b} \end{bmatrix}^T \in \mathbb{R}^{n_\theta}, \tag{2.4a}$$
$$\bar{\theta} \triangleq \begin{bmatrix} \bar{a}^T & b^T \end{bmatrix}^T = \begin{bmatrix} 1 & \theta^T \end{bmatrix}^T \in \mathbb{R}^{n_\theta+1}, \tag{2.4b}$$

where $n_\theta = n_a + n_b$. Using a linear regression formulation, an alternative description of (2.1)-(2.3) is given by

$$\bar{\varphi}_{0_k}^T \bar{\theta} = 0, \tag{2.5a}$$
$$\bar{\varphi}_k = \bar{\varphi}_{0_k} + \tilde{\bar{\varphi}}_k, \tag{2.5b}$$

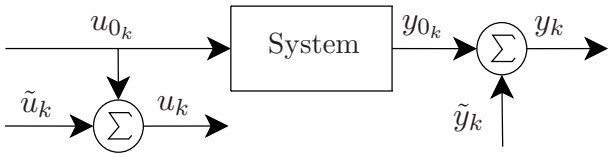


Figure 2.1: Errors-in-variables setup.

where

$$\varphi_{0_k} \triangleq [-y_{0_{k-1}} \quad ... \quad -y_{0_{k-n_a}} \quad u_{0_{k-1}} \quad ... \quad u_{0_{k-n_b}}]^T, \tag{2.6a}$$

$$\bar{\varphi}_{0_k} \triangleq [-y_{0_k} \quad \varphi_{0_k}^T]^T, \tag{2.6b}$$

$$\varphi_k \triangleq [-y_{k-1} \quad ... \quad -y_{k-n_a} \quad u_{k-1} \quad ... \quad u_{k-n_b}]^T, \tag{2.6c}$$

$$\bar{\varphi}_k \triangleq [-y_k \quad \varphi_k^T]^T, \tag{2.6d}$$

$$\tilde{\varphi}_k \triangleq [-\tilde{y}_{k-1} \quad ... \quad -\tilde{y}_{k-n_a} \quad \tilde{u}_{k-1} \quad ... \quad \tilde{u}_{k-n_b}]^T, \tag{2.6e}$$

$$\tilde{\bar{\varphi}}_k \triangleq [-\tilde{y}_k \quad \tilde{\varphi}_k^T]^T. \tag{2.6f}$$

In addition, it is often convenient to divide the regression vector into two parts corresponding to the outputs and inputs, respectively. This yields

$$\varphi_k^T \triangleq \begin{bmatrix} \varphi_{y_k}^T & \varphi_{u_k}^T \end{bmatrix}^T, \tag{2.7a}$$

$$\bar{\varphi}_k^T \triangleq \begin{bmatrix} \bar{\varphi}_{y_k}^T & \varphi_{u_k}^T \end{bmatrix}^T. \tag{2.7b}$$

For the white noise case, it is convenient to introduce the augmented parameter vector

$$\vartheta \triangleq \begin{bmatrix} \theta^T & \sigma^T \end{bmatrix}^T \in \mathbb{R}^{n_\theta+2}, \tag{2.8}$$

which comprises both the system and the noise parameter vectors, where the latter is given by

$$\sigma \triangleq \begin{bmatrix} \sigma_{\tilde{y}} & \sigma_{\tilde{u}} \end{bmatrix}^T \in \mathbb{R}^2, \tag{2.9}$$

with $\sigma_{\tilde{y}}$ and $\sigma_{\tilde{u}}$ denoting the variance of the output noise and input noise, respectively.

Within this thesis, cross- and auto-covariance functions are defined by

$$r_{cd}(\tau) \triangleq E\left[c_k d_{k-\tau}\right], \tag{2.10a}$$

$$r_c(\tau) \triangleq E\left[c_k c_{k-\tau}\right]. \tag{2.10b}$$

where c_k and d_k denote two arbitrary zero mean stochastic processes and where $E[\cdot]$ denotes the expected value operator. The matrices and vectors comprising these covariance elements are denoted as Σ and ξ, respectively. Consequently, the cross/auto-covariance matrices of two arbitrary random vectors v_k and w_k are denoted

$$\Sigma_{vw} \triangleq E\left[v_k w_k^T\right], \tag{2.11a}$$

$$\Sigma_v \triangleq E\left[v_k v_k^T\right], \tag{2.11b}$$

whilst the cross-covariance vector between an arbitrary random vector v_k and a scalar

stochastic process c_k is denoted

$$\xi_{vc} \triangleq E\left[v_k c_k\right] \qquad \text{(column vector)}, \tag{2.12a}$$

$$\xi_{cv} \triangleq E\left[c_k v_k^T\right] \qquad \text{(row vector)'}. \tag{2.12b}$$

The corresponding estimated sample covariance elements comprising scalars, vectors and matrices are denoted in a similar manner, given by

$$\hat{r}_{cd}(\tau) \triangleq \frac{1}{k}\sum_{i=1}^{k} c_i d_{i-\tau}, \tag{2.13a}$$

$$\hat{r}_{c}(\tau) \triangleq \frac{1}{k}\sum_{i=1}^{k} c_i c_{i-\tau}, \tag{2.13b}$$

$$\hat{\Sigma}_{vw} \triangleq \frac{1}{k}\sum_{i=1}^{k} v_i w_i^T, \tag{2.13c}$$

$$\hat{\Sigma}_{v} \triangleq \frac{1}{k}\sum_{i=1}^{k} v_i v_i^T, \tag{2.13d}$$

$$\hat{\xi}_{vc} \triangleq \frac{1}{k}\sum_{i=1}^{k} v_i c_i, \tag{2.13e}$$

$$\hat{\xi}_{cv} \triangleq \frac{1}{k}\sum_{i=1}^{k} c_i v_i^T. \tag{2.13f}$$

If the data is not equally weighted, a weighted arithmetic mean is to be computed, which is denoted by

$$\bar{\Sigma}_{vw} \triangleq \gamma_k \sum_{i=1}^{k} \beta_k^i v_i w_i^T, \tag{2.14}$$

where γ_k is an appropriately chosen normalising gain and β_k^i is the weighting of the ith measurement at time k.

In the white noise case, the auto-covariance matrix $\Sigma_{\tilde{\varphi}}$ corresponding to the noisy part of the regression vector (2.6e) is diagonal, dependent on $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$ and given as

$$\Sigma_{\tilde{\varphi}}(\sigma) \triangleq \begin{bmatrix} \sigma_{\tilde{y}} I_{n_a} & 0 \\ 0 & \sigma_{\tilde{u}} I_{n_b} \end{bmatrix}. \tag{2.15}$$

For the coloured output noise case, introduce the vector

$$\rho_{\tilde{y}} \triangleq \begin{bmatrix} r_{\tilde{y}}(0) & r_{\tilde{y}}(1) & \cdots & r_{\tilde{y}}(n_a) \end{bmatrix}^T \in \mathbb{R}^{n_a+1}, \tag{2.16}$$

which comprises the auto-correlation elements of $\tilde{y}_k$, noting that $r_{\tilde{y}}(0) = \sigma_{\tilde{y}}$. The

augmented parameter vector for this case becomes

$$\Theta \triangleq \begin{bmatrix} a_1 & ... & a_{n_a} & b_1 & ... & b_{n_b} & \rho_{\tilde{y}} & \sigma_{\tilde{u}} \end{bmatrix}^T \in \mathbb{R}^{2n_a+n_b+2}. \tag{2.17}$$

2.3 Mathematical tools

Some mathematical concepts are repeatedly used in the thesis, and, therefore, these are reviewed within this section, for future reference.

2.3.1 Stationary iterative methods for least squares

The least squares (LS) problem can be solved in an iterative way by making use of a suitable matrix splitting. These so-called stationary iterative methods are commonly used for large sparse LS problems where an initial approximate solution is iteratively improved until an appropriate stopping criterion is satisfied (Björck 1996, Ch. 7). Based on the normal equations

$$\Sigma\theta = \xi, \tag{2.18}$$

where Σ is a matrix, ξ a vector and θ a parameter vector of interest, the underlying idea of these methods is to split the matrix Σ into $\Sigma = \Sigma_1 - \Sigma_2$ and solve the resulting system of equations in an iterative manner. This gives

$$\hat{\theta}_{k+1} = \Sigma_1^{-1}\Sigma_2\hat{\theta}_k + \Sigma_1^{-1}\xi. \tag{2.19}$$

Naturally, the splitting is chosen such that the inverse computation of Σ_1 is easy. Note, however, that it is stated in (Björck 1996) that the main weakness of such iterative methods is their "poor robustness and often narrow range of applicability." From (2.19) it is clear that the iteration converges for all initial vectors $\hat{\theta}_0$ if all eigenvalues of $\Sigma_1^{-1}\Sigma_2$ are within the unit circle. The principle of stationary iterative methods finds several applications within this thesis, when it is desired to derive a recursive estimation procedure without computing the (pseudo) inverse of Σ. A thorough treatment of this concept can be found in (Björck 1996, Ch. 7).

2.3.2 Variable projection algorithm

The system identification problem frequently reduces to a nonlinear least squares (NLS) problem, given by

$$\min_{\vartheta} \ \|r(\vartheta)\|_2^2, \tag{2.20}$$

which aims to find the global minimiser of the sum of squares of m nonlinear residual functions $r_i(\vartheta)$, $i = 1, ..., m$. The NLS problem is denoted separable, if the solution

vector ϑ can be partitioned, e.g.

$$\vartheta = \begin{bmatrix} \theta^T & \sigma^T \end{bmatrix}^T \tag{2.21}$$

such that the subproblem

$$\min_{\theta} \|r(\theta, \sigma)\|_2^2 \tag{2.22}$$

"is easy to solve" (Björck 1996). For instance, if $r(\theta, \sigma)$ is linear in θ the residual can be expressed as

$$r(\theta, \sigma) = F(\sigma)\theta - f(\sigma), \tag{2.23}$$

where $F(\sigma)$ is a $m \times n_\theta$ matrix and $f(\sigma)$ is a vector of dimension m. Hence, the minimum norm solution of the subproblem (2.22) is given by

$$\theta(\sigma) = F(\sigma)^\dagger f(\sigma), \tag{2.24}$$

where $F(\sigma)^\dagger$ denotes the Moore-Penrose pseudo inverse. Consequently, (2.24) can be substituted into (2.23), which allows (2.20) to be expressed as

$$\begin{aligned} \min_{\sigma} \ \|r(\sigma)\|_2^2 &= \min_{\sigma} \ \|f(\sigma) - F(\sigma)\theta(\sigma)\|_2^2 \\ &= \min_{\sigma} \ \left\| \left[I - F(\sigma)F(\sigma)^\dagger \right] f(\sigma) \right\|_2^2, \end{aligned} \tag{2.25}$$

where $F(\sigma)F(\sigma)^\dagger$ is the orthogonal projector on the range of $F(\sigma)$. Therefore, algorithms based on (2.25) are also referred to as variable projection algorithms or separable NLS. A standard reference is given by (Golub & Pereyra 1973) whilst a more recent overview is given by (Golub & Pereyra 2002). The variable projection approach has been exploited for the identification of EIV systems using the so-called extended bias compensating least squares (EBCLS) approach which has been developed in (Ekman 2005), where a consistency analysis has also been provided (see Section 2.4.2 for a review of the EBCLS algorithm).

The variable projection technique exhibits the following properties:

- The global minima of the functionals $r(\theta, \sigma)$ and $r(\sigma)$ coincide and have the same values, i.e. $\|r(\hat{\theta}, \hat{\sigma})\|_2^2 = \|r(\hat{\sigma})\|_2^2$ (Theorem 2.1 in Golub & Pereyra 1973).
- The dimension of the resulting minimisation problem is reduced. This also implies that fewer initial parameter values are required to be specified which might also reduce the problem of becoming trapped within a local minimum.
- In comparison to the direct minimisation of (2.20), the variable projection algorithm is always able to converge in less iterations (Golub & Pereyra 2002).

Description	Operation	Flops
Scalar addition	$c + c$	1
Scalar multiplication	c^2	1
Outer product	yy^T	p^2
Inner product	$y^T y$	$2p$
Addition	$M + M$	pr
Scalar multiplication	cM	pr
Matrix product	LM	$2pvr$
Matrix-vector product	Ly	$2pv$
Squared vector-norm	$\|\|y\|\|_2^2$	$2p$
Square root	$\sqrt{c}$	8
Inverse (general)	S^{-1}	$O(s^3)$
Least eigenvalue (general)	$\lambda_{\min}(S)$	$O(s^3)$

Table 2.1: Some matrix operations and associated number of flops, where $y \in \mathbb{R}^p$, $M \in \mathbb{R}^{p \times r}$, $c \in \mathbb{R}$, $S \in \mathbb{R}^{s \times s}$ is symmetric and $L \in \mathbb{R}^{v \times p}$.

- Although the reduced functional is more complex, the total computing time is smaller for a large class of problems (Golub & Pereyra 2002).

Note that the problem simplifies if the nonlinear set of functions $r(\vartheta)$ is not only linear in θ, but also linear in σ, which means one can express

$$r(\theta, \sigma) = G(\theta)\sigma - g(\theta) \tag{2.26}$$

with $G(\theta) \in \mathbb{R}^{m \times n_\sigma}$ and $g(\theta) \in \mathbb{R}^m$. Such a situation is also referred to as bilinear parametrisation[1] (Ljung 1999, cf. p. 335).

2.3.3 Measuring computational complexity

In order to obtain an approximate measure for the computational costs of an algorithm the floating point operations (flops) are counted as described in (Golub & Van Loan 1996, p. 18). Some matrix operations and corresponding number of flops, which are utilised in the subsequent development, are summarised in Table 2.1. Flop counting is only a crude measure for the computational burden and the order of the complexity is frequently given using the 'big-O-notation', e.g. $O(n^2)$ for quadratic complexity, where only the dominant factors are considered, i.e. less significant contributions are dismissed. Although it is only approximate, flop counting is a rather convenient way to measure the computational costs of an algorithm, since it is quite general due to its independence of the underlying hardware, programming language and implementation. Alternatively, it is possible to measure the absolute computation time on a given machine using a certain programming environment.

Note that the computational complexity evaluated using by the 'big-O-notation' considers the asymptotic behaviour for very large model orders. In the context of

[1] Not to be confused with bilinear system representations.

system identification, however, mainly low model orders are of interest, hence, the absolute computation time for such low orders will also be investigated via numerical simulations.

2.4 System identification

This section reviews EIV system identification techniques which form the basis for the development in the subsequent chapters. In the white noise case, the EIV identification problem can be formulated as follows.

Problem 2.1. Given k measured input-output samples, denoted,

$$Z^k \triangleq \{u_i, y_i\}_{i=1}^k, \tag{2.27}$$

determine an estimate of the augmented parameter vector

$$\vartheta = \begin{bmatrix} a_1 & \dots & a_{n_a} & b_1 & \dots & b_{n_b} & \sigma_{\tilde{y}} & \sigma_{\tilde{u}} \end{bmatrix}^T. \tag{2.28}$$

If not stated otherwise, assumptions concerning the system, the input and the noise sequences are given as follows.

System assumptions:

AS1 The dynamic system is asymptotically stable, i.e. $A(q^{-1})$ has all zeros inside the unit circle.

AS2 All system modes are observable and controllable, i.e. $A(q^{-1})$ and $B(q^{-1})$ have no common factors.

AS3 The polynomial degrees n_a and n_b are known a priori with $n_b \leq n_a$.

Input assumptions:

AI1 The true input u_{0_k} is a zero-mean ergodic process and is persistently exciting of sufficiently high order.

Noise assumptions:

AN1 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are zero-mean, ergodic, white noises with unknown variances, denoted $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, respectively, i.e.

$$\sigma_{\tilde{u}}\delta_{kl} \triangleq E\left[\tilde{u}_k\tilde{u}_l\right], \tag{2.29a}$$

$$\sigma_{\tilde{y}}\delta_{kl} \triangleq E\left[\tilde{y}_k\tilde{y}_l\right]. \tag{2.29b}$$

AN2 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are mutually uncorrelated and also uncorrelated with both u_{0_k} and y_{0_k}.

Within Assumption **AN1**, δ_{kl} denotes the Kronecker delta. Note that the assumption of $n_b \leq n_a$ within **AS3** is common when dealing with control problems. In contrast, within a system identification context, such an additional assumption might not be required.

2.4.1 Bias compensating least squares

It is well known that the parameter estimates are asymptotically biased when LS is utilised to solve the EIV identification problem. However, if the variances of the input and output measurement noise sequences are available, it is possible to compensate for the bias in the LS solution via

$$\hat{\theta} = \left[\Sigma_{\varphi} - \Sigma_{\tilde{\varphi}}(\sigma)\right]^{-1} \xi_{\varphi y}. \tag{2.30}$$

Such bias compensating least squares (BCLS) approaches are well known in the system identification literature and an early reference is given by (Stoica & Söderström 1982), where this principle is used to identify output error models. By utilising the natural matrix splitting $\Sigma_{\varphi} - \Sigma_{\tilde{\varphi}}(\sigma)$, it is possible to solve (2.30) via a stationary iterative LS method (cf. Section 2.3.1) by expressing (2.30) as

$$\begin{aligned} & \Sigma_{\varphi}\hat{\theta}_k = \xi_{\varphi y} + \Sigma_{\tilde{\varphi}}(\sigma)\hat{\theta}_{k-1} \\ \Leftrightarrow \quad & \hat{\theta}_k = \hat{\theta}_k^{\mathrm{LS}} + \Sigma_{\varphi}^{-1}\Sigma_{\tilde{\varphi}}(\sigma)\hat{\theta}_{k-1}, \end{aligned} \tag{2.31}$$

where $\hat{\theta}_k^{\mathrm{LS}}$ denotes the LS estimate. In order to obtain a corresponding recursive scheme, the covariance matrices are required to be updated at each time instance, whilst the LS estimate can be computed via a standard recursive least squares (RLS) algorithm (Sagara & Wada 1977, Ding et al. 2006). Such a recursive BCLS (RBCLS) scheme is given by

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{LS}} + P_k\Sigma_{\tilde{\varphi}}(\sigma)\hat{\theta}_{k-1}, \tag{2.32}$$

where the need for a matrix inversion is conveniently avoided by utilising the appropriately scaled error covariance matrix, denoted P_k, of the RLS algorithm, which can be computed via the matrix inversion lemma (cf. Ljung 1999, Ch. 11). This RBCLS approach forms the basis for a number of algorithms, which are developed within this thesis.

It is interesting to note that an iterative BCLS approach as outlined above also forms the basis of the so-called bias eliminating least squares (BELS) techniques (Zheng & Feng 1989, Zheng 1998, Zheng 1999, Zheng 2000) for the identification of EIV systems.

2.4.2 Extended bias compensating least squares

As its name indicates, the extended bias compensating least squares (EBCLS) technique (Ekman 2005) is a generalisation of the bias compensating least squares approach, which has been introduced in Section 2.4.1. Rather than making use of the standard LS normal equations, an instrumental variable (IV) approach with an arbitrary instrument vector, denoted $z_k \in \mathbb{R}^{n_z}$, where $n_z \geq n_\theta$ is considered. The EBCLS estimate is obtained by solving

$$[\Sigma_{z\varphi} - \Sigma_{\tilde{z}\tilde{\varphi}}(\sigma)]\,\theta = \xi_{zy} - \xi_{\tilde{z}\tilde{y}}(\sigma), \tag{2.33}$$

where the individual quantities are defined by (2.11)-(2.12). Note, from (2.33), that depending on the particular choice of z_k, not only the covariance matrix on the left hand side, but also the covariance vector on the right needs to be compensated, in order to remove the asymptotic bias of the IV estimate. Of course it is possible to select z_k, such that no correlation between the instruments and φ_k, y_k exists. However, since the objective is to estimate not only θ but also σ, the instruments are usually chosen such that (2.33) depends on σ, i.e. $\Sigma_{\tilde{z}\tilde{\varphi}}(\sigma) \neq 0$ and/or $\zeta_{\tilde{z}\tilde{y}}(\sigma) \neq 0$. The noise variance estimates can then be determined by minimising the sum of the squared residuals

$$\hat{\sigma} = \arg\min_{\sigma} \; \|[\Sigma_{z\varphi} - \Sigma_{\tilde{z}\tilde{\varphi}}(\sigma)]\,\theta - \xi_{zy} + \xi_{\tilde{z}\tilde{y}}(\sigma)\|_2^2\,, \tag{2.34}$$

which is a NLS problem. Note that the resulting NLS problem is separable with respect to θ and σ. Consequently, it can be solved by means of the variable projection algorithm (cf. Section 2.3.2), where

$$F(\sigma) = \Sigma_{z\varphi} - \Sigma_{\tilde{z}\tilde{\varphi}}(\sigma), \tag{2.35a}$$

$$f(\sigma) = \xi_{zy} - \xi_{\tilde{z}\tilde{y}}(\sigma). \tag{2.35b}$$

Once $\hat{\sigma}$ has been determined, $\hat{\theta}$ is obtained by solving (2.33), where σ is replaced by $\hat{\sigma}$.

2.4.3 Dynamic Frisch scheme for white noise

One particularly interesting approach for the identification of dynamical EIV systems, which yields estimates of the model parameters as well as the measurement noise variances, is the so-called Frisch scheme (Beghelli et al. 1990, Söderström 2007*b*). It was originally developed to treat static algebraic regression problems (Frisch 1934) without making any assumptions on the relative amount of noise on the variables. These rather loose constraints on the required a priori knowledge yield a whole family of solutions. The extension of the Frisch scheme to deal with dynamical multiple-input single-output (MISO) LTI systems (Beghelli et al. 1990) leads theoretically to a single solution. In practice, however, a model selection criterion is required to be utilised, in order to choose an 'optimal' solution from a set of possible Frisch scheme models. Three dif-

ferent options are discussed in (Hong et al. 2007), whilst the statistical accuracy of the Frisch scheme has been analysed in (Söderström 2007*a*). Recent work (Hong & Söderström 2008) has also established the connection of the Frisch scheme with the BELS algorithms (Zheng & Feng 1989, Zheng 1998) as well as the EBCLS method (Ekman 2005).

Frisch scheme approach

Premultiplying (2.5a) with $\bar{\varphi}_{0_k}$ and taking the expected value leads to an alternative system description

$$\Sigma_{\bar{\varphi}_0}\bar{\theta} = 0, \tag{2.36}$$

where $\Sigma_{\bar{\varphi}_0} \in \mathbb{R}^{(n_\theta+1)\times(n_\theta+1)}$ is the noise-free covariance matrix (cf. (2.11)), which is singular positive semidefinite, with $\text{rank}(\Sigma_{\bar{\varphi}_0}) = n_a + n_b$ (i.e. rank-one deficient). Due to the stated assumptions, the noise-free covariance matrix can be decomposed into

$$\Sigma_{\bar{\varphi}_0} = \Sigma_{\bar{\varphi}} - \Sigma_{\tilde{\bar{\varphi}}}(\sigma), \tag{2.37}$$

where $\Sigma_{\bar{\varphi}}$ is the covariance matrix of the noisy data whilst the noise covariance matrix is given by

$$\Sigma_{\tilde{\bar{\varphi}}}(\sigma) = \begin{bmatrix} \sigma_{\tilde{y}} I_{n_a+1} & 0 \\ 0 & \sigma_{\tilde{u}} I_{n_b} \end{bmatrix}. \tag{2.38}$$

Note that, in the noisy case, the covariance matrix $\Sigma_{\bar{\varphi}}$ is generally of full rank, hence, the Frisch scheme identification problem can be re-expressed as follows.

Problem 2.2. Given the data covariance matrix $\Sigma_{\bar{\varphi}}$ of noisy observations, determine the noise covariance matrix $\Sigma_{\tilde{\bar{\varphi}}}$ such that

$$\Sigma_{\bar{\varphi}_0} = \Sigma_{\bar{\varphi}} - \hat{\Sigma}_{\tilde{\bar{\varphi}}}(\sigma) \geq 0 \tag{2.39a}$$

and

$$\det(\Sigma_{\bar{\varphi}_0}) = 0. \tag{2.39b}$$

This means that solutions for the values of $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$ are searched for, such that the resulting matrix $\Sigma_{\bar{\varphi}_0}$ becomes singular. Equation (2.39) essentially defines the core of the Frisch scheme since these properties distinguish this approach from other EIV identification techniques. For future reference, the following definition is introduced.

Definition 2.1 (Frisch-character). An estimated parameter vector $\hat{\vartheta} = [\hat{\theta}^T \ \hat{\sigma}^T]^T$ is said

to satisfy the Frisch-character if

$$\det\left(\hat{\Sigma}_{\tilde{\varphi}} - \Sigma_{\tilde{\hat{\varphi}}}(\hat{\sigma})\right) = 0. \tag{2.40}$$

Problem 2.2 does not yield a unique solution, rather the set of admissible solutions $(\sigma_{\tilde{u}}, \sigma_{\tilde{y}})$ which satisfies (2.39) defines a convex curve in the first quadrant of the noise space $\mathbb{R}^2$, which is shown in Figure 2.2, for an arbitrary example. A further consequence

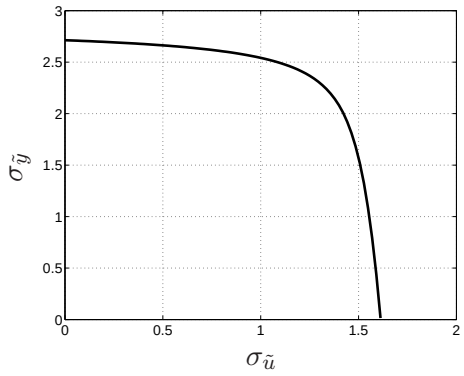


Figure 2.2: Convex curve in the noise space characterising the Frisch scheme.

of this approach is that each point on the convex curve can be uniquely mapped into the parameter space $\mathbb{R}^{n_a+n_b}$. This means that a particular solution of the EIV identification problem can be characterised by the input measurement noise variance only, since a given $\sigma_{\tilde{u}}$ uniquely defines $\sigma_{\tilde{y}}$, hence θ (Beghelli et al. 1990). The mappings within the Frisch scheme are consequently given by

$$\sigma_{\tilde{u}} \mapsto \sigma_{\tilde{y}}, \tag{2.41a}$$

$$(\sigma_{\tilde{u}}, \sigma_{\tilde{y}}) \mapsto \theta. \tag{2.41b}$$

In order to express the Frisch scheme relationships with some mathematical rigour, the first mapping, defining the relationship between $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, is given by the function

$$f : [0, \sigma_{\tilde{u}}^{\max}] \times \mathcal{Z} \to [0, \sigma_{\tilde{y}}^{\max}], \tag{2.42}$$

where $\mathcal{Z}$ denotes the family of sets comprising all possible data sets, i.e. $Z^k \in \mathcal{Z}$, whilst $\times$ denotes the Cartesian product (set of all possible ordered pairs). The quantities $\sigma_{\tilde{u}}^{\max}$ and $\sigma_{\tilde{y}}^{\max}$ are the maximal admissible values for $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, respectively, which correspond to the intersections of the convex curve with the horizontal and vertical axes in Figure 2.2. Equivalently, it is possible to obtain $\sigma_{\tilde{u}}$ based on $\sigma_{\tilde{y}}$ and the data Z^k which might be formulated as

$$g : [0, \sigma_{\tilde{y}}^{\max}] \times \mathcal{Z} \to [0, \sigma_{\tilde{u}}^{\max}]. \tag{2.43}$$

Finally, the relationship between $(\sigma_{\tilde{u}}, \sigma_{\tilde{y}})$ and θ can be expressed as

$$h : [\sigma_{\tilde{u}}, \sigma_{\tilde{y}}] \times \mathcal{Z} \to D_{\mathcal{M}} \subset \mathbb{R}^{n_\theta}, \tag{2.44}$$

where $D_{\mathcal{M}}$ is a subset of $\mathbb{R}^{n_\theta}$, such that $\theta \in D_{\mathcal{M}}$. The function h is defined by solving the consistent set of normal equations

$$\left(\Sigma_\varphi - \begin{bmatrix} \sigma_{\tilde{y}} I_{n_a} & 0 \\ 0 & \sigma_{\tilde{u}} I_{n_b} \end{bmatrix} \right) \theta = \xi_{\varphi y}. \tag{2.45}$$

Note that if only one particular data set is regarded, i.e. Z^k is one particular realisation of its corresponding random sequence (Z^k is kept constant), the above functions f, g and h are bijective[2] and $f^{-1} = g$, where f^{-1} denotes the inverse relation.

Relation between $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$

The key question that is now addressed is: How are f and g defined? i.e. how can the output noise variance be determined knowing the input noise variance or vice versa. Utilising the Schur complement, the (nonlinear) relationship is given by (Beghelli et al. 1990, Söderström 2007*a*)

$$\sigma_{\tilde{u}} = f(\sigma_{\tilde{y}}, Z^k) = \lambda_{\min} \left(\Sigma_{\varphi_u} - \Sigma_{\varphi_u \bar{\varphi}_y} \left[\Sigma_{\bar{\varphi}_y} - \hat{\sigma}_{\tilde{y}} I_{n_a+1} \right]^{-1} \Sigma_{\bar{\varphi}_y \varphi_u} \right), \tag{2.46a}$$

or equivalently, the inverse relation is given by

$$\sigma_{\tilde{y}} = g(\sigma_{\tilde{u}}, Z^k) = \lambda_{\min} \left(\Sigma_{\bar{\varphi}_y} - \Sigma_{\bar{\varphi}_y \varphi_u} \left[\Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b} \right]^{-1} \Sigma_{\varphi_u \bar{\varphi}_y} \right), \tag{2.46b}$$

where $\lambda_{\min}$ denotes the minimum eigenvalue operator and the individual block matrices are defined by

$$\Sigma_{\bar{\varphi}} = \begin{bmatrix} \Sigma_{\bar{\varphi}_y} & \Sigma_{\bar{\varphi}_y \varphi_u} \\ \Sigma_{\varphi_u \bar{\varphi}_y} & \Sigma_{\varphi_u} \end{bmatrix} \tag{2.47}$$

with $\Sigma_{\bar{\varphi}_y} \in \mathbb{R}^{n_a+1 \times n_a+1}$ and $\Sigma_{\varphi_u} \in \mathbb{R}^{n_b \times n_b}$ (also recall the definitions (2.7) and (2.11)). In the remainder of this thesis Equations (2.46a) and (2.46b) are also referred to as the $\lambda_{\min}$-equations. In addition, (2.46) provides explicit expressions for the maximal admissible values for $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, respectively. By setting $\sigma_{\tilde{y}} = 0$ in (2.46a) and $\sigma_{\tilde{u}} = 0$ in (2.46b) these quantities are given by

$$\sigma_{\tilde{u}}^{\max} = \lambda_{\min} \left(\Sigma_{\varphi_u} - \Sigma_{\varphi_u \bar{\varphi}_y} \Sigma_{\bar{\varphi}_y}^{-1} \Sigma_{\bar{\varphi}_y \varphi_u} \right), \tag{2.48a}$$

$$\sigma_{\tilde{y}}^{\max} = \lambda_{\min} \left(\Sigma_{\bar{\varphi}_y} - \Sigma_{\bar{\varphi}_y \varphi_u} \Sigma_{\varphi_u}^{-1} \Sigma_{\varphi_u \bar{\varphi}_y} \right). \tag{2.48b}$$

[2] A function is bijective if it is injective and surjective. A bijection is sometime referred to as one-to-one correspondence.

Model selection - determination of $\sigma_{\tilde{u}}$

In order to solve the identification problem, $n_\theta + 2$ equations are required to determine the n_θ model parameters as well as $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$. So far, only $n_\theta + 1$ equations are established: the n_θ normal equations given by (2.45) plus the $\lambda_{\min}$-equation (2.46b)[3]. Consequently, there is one degree of freedom remaining and an additional equation is required in order to select one point from the convex curve corresponding to a certain model. Four different methods are known within the literature and each of them gives rise to a different Frisch scheme algorithm. These are:

Extended model criterion (EM): The $\lambda_{\min}$-equation is evaluated for the nominal and an extended model structure within the range $0 \leq \sigma_{\tilde{u}} \leq \sigma_{\tilde{u}}^{\max}$, where the $A(q^{-1})$-polynomial or the $B(q^{-1})$ polynomial (or both) are extended with fictitious zero valued parameters. This results in two curves (corresponding to the nominal and the extended model) in the noise plane that theoretically are tangential to each other at a unique point, which corresponds to the 'true' noise variance values (Beghelli et al. 1990). The algorithm is denoted as Frisch-EM.

Covariance match criterion (CM): The statistical properties of the residuals computed from the system are compared with those predicted from a certain model (Diversi, Guidorzi & Soverini 2003*b*, Söderström 2007*a*). This algorithm is denoted as Frisch-CM.

Yule-Walker criterion (YW): The set of high order Yule-Walker equations can be exploited, which is equivalent to the utilisation of an additional IV estimator (cf. e.g. Söderström & Mahata 2002, Söderström et al. 2002) that assesses the quality of the admissible solutions (Diversi et al. 2006). The corresponding algorithm is denoted as Frisch-YW.

The model selection cost function proposed in (Diversi et al. 2006) is given by

$$V_k(\sigma_{\tilde{u}}) = \frac{1}{2}\|\hat{\Sigma}^k_{\zeta\bar{\varphi}}\bar{\theta}\|_2^2. \tag{2.49}$$

It basically corresponds to the Euclidean norm of the residuals given a certain value for θ which is determined by means of an additional IV estimator with delayed (or time shifted) inputs as instruments. In (Diversi et al. 2006), the instrument vector is given by

$$\zeta_k \triangleq \begin{bmatrix} u_{k-n_b-1} & \dots & u_{k-n_b-n_\zeta} \end{bmatrix}^T \in \mathbb{R}^{n_\zeta}, \tag{2.50}$$

where the instrument dimension, denoted $n_\zeta \geq n_a + n_b + 1$, is user specified. Since a certain $\sigma_{\tilde{u}}$ uniquely defines θ, $V_k(\sigma_{\tilde{u}})$ can be minimised with respect to

[3] Usually, (2.46b) is chosen, but it would also be possible to use (2.46a).

$\sigma_{\tilde{u}}$ giving

$$\hat{\sigma}_{\tilde{u}} = \arg \min_{\sigma_{\tilde{u}}} V_k(\sigma_{\tilde{u}}). \tag{2.51}$$

Note that in the asymptotic case the true parameter vector satisfies

$$\Sigma_{\zeta\bar{\varphi}}\bar{\theta} = 0. \tag{2.52}$$

Extended bias-compensating normal equations (EC): An EBCLS estimator assesses the quality of the admissible solutions. Whilst ζ_k for the YW criterion has to be chosen, such that the instruments are uncorrelated with $\tilde{\bar{\varphi}}_k$ (in order to obtain a consistent IV estimator), it is possible to choose a general instrument vector which does not satisfy $E[\zeta_k\tilde{\bar{\varphi}}_k^T] = 0$ and then to compensate for the resulting bias. This algorithm, which is denoted as Frisch-EC, has recently been proposed in (Larkowski, Linden, Vinsonneau & Burnham 2008)[4]. The general model selection criterion is given by

$$V_{\mathrm{EC}}(\sigma_{\tilde{u}}) \triangleq \|\xi_{z_2y} - \xi_{\tilde{z}_2\tilde{y}} - \big(\Sigma_{z_2\varphi} - \Sigma_{\tilde{z}_2\tilde{\varphi}}\big)\big(\Sigma_{z_3\varphi} - \Sigma_{\tilde{z}_3\tilde{\varphi}}\big)^{\dagger}\big(\xi_{z_3y} - \xi_{\tilde{z}_3\tilde{y}}\big)\|_2^2, \tag{2.53}$$

where z_{2_k} and z_{3_k} denote arbitrary instrument vectors whilst $\tilde{z}_{2_k}$ and $\tilde{z}_{3_k}$ denote the corresponding noise conributions[5]. In (Larkowski et al. 2008), the instrument vectors

$$z_{1_k} = \varphi_k \tag{2.54a}$$

$$z_{2_k} = z_{3_k} = \begin{bmatrix} -y_k & \cdots & -y_{k-n_a-m} & u_k & \cdots & u_{k-n_b-m} \end{bmatrix}^T \tag{2.54b}$$

have been utilised, which basically corresponds to the choice of instruments that has been suggested in (Ekman 2005) for the EBCLS method. The quantity $m \geq 0$ is chosen by the user and denotes the number of supplemental normal equations. Note that the YW model selection criterion can be considered as being a subset of the more general EC criterion.

A comparison of the EM, CM and YW model selection criteria is given in (Hong et al. 2007). Note that all of the different Frisch scheme approaches outlined above require the solution of a one-dimensional optimisation problem. The search for the optimal solution can be constrained using the maximal admissible values given in (2.48). Within the remainder of this thesis, attention is restricted to the YW criterion.

[4]The author is an advisor for this research programme.

[5]The notation used here for the instrument vectors is in agreement with that used in (Larkowski et al. 2008). There, the variable z_{1_k} is used in to denote the instrument vector which defines the set of normal equations which are used to determine θ. Whilst this yields a more general setup, $z_{1_k} = \varphi_k$ holds in the particular case of the Frisch scheme.

Summary

The different Frisch scheme approaches may be summarised in the following steps:

1. Find an 'optimal' $\hat{\sigma}_{\tilde{u}}$ in the sense of a chosen model criteria outlined above.
2. Compute $\hat{\sigma}_{\tilde{y}}$ using

$$\hat{\sigma}_{\tilde{y}} = f\left(\hat{\sigma}_{\tilde{u}}, Z^k\right). \tag{2.55}$$

3. Determine the parameter vector via BCLS

$$\hat{\theta} = h\left(\hat{\sigma}_{\tilde{u}}, \hat{\sigma}_{\tilde{y}}, Z^k\right). \tag{2.56}$$

Note that $\sigma_{\tilde{u}}$ in Step 1 requires an optimisation, which involves the determination θ and $\sigma_{\tilde{y}}$ for a given $\sigma_{\tilde{u}}$ at each iteration step. Hence, each iteration also requires the solution of (2.46a) and (2.45) to be computed.

2.4.4 Dynamic Frisch scheme for coloured output noise

The dynamic Frisch scheme presented in (Beghelli et al. 1990, Söderström 2007*a*) assumes that the additive disturbances on the system input and output are white. Such an assumption, however, can be rather restrictive since the output noise often not solely consists of measurement uncertainties, but also aims to account for process disturbances, which are usually correlated in time. In order to overcome this shortcoming, the Frisch scheme has recently been extended to the coloured output noise case (Söderström 2006, Söderström 2008).

Problem statement

If the output noise sequence $\tilde{y}_k$ is correlated in time, the bias compensated normal equations (2.45) become

$$\begin{aligned} & \Sigma_{\varphi_0}\theta = \xi_{\varphi_0 y} \\ \Leftrightarrow \quad & \left(\Sigma_{\varphi} - \Sigma_{\tilde{\varphi}}\right)\theta = \xi_{\varphi y} - \xi_{\tilde{\varphi}\tilde{y}}, \end{aligned} \tag{2.57}$$

or in block matrix form

$$\left(\begin{bmatrix} \Sigma_{\varphi_y} & \Sigma_{\varphi_y\varphi_u} \\ \Sigma_{\varphi_u\varphi_y} & \Sigma_{\varphi_u} \end{bmatrix} - \begin{bmatrix} \Sigma_{\tilde{\varphi}_y} & 0 \\ 0 & \sigma_{\tilde{u}} I_{n_b} \end{bmatrix}\right)\theta = \begin{bmatrix} \xi_{\varphi_y y} \\ \xi_{\varphi_u y} \end{bmatrix} - \begin{bmatrix} \xi_{\tilde{\varphi}_y\tilde{y}} \\ 0 \end{bmatrix}, \tag{2.58}$$

where

$$\Sigma_{\tilde{\varphi}_y} = \begin{bmatrix} r_{\tilde{y}}(0) & \cdots & r_{\tilde{y}}(n_a-1) \\ \vdots & \ddots & \vdots \\ r_{\tilde{y}}(n_a-1) & \cdots & r_{\tilde{y}}(0) \end{bmatrix} \tag{2.59}$$

is a symmetric Toeplitz matrix, whilst

$$\xi_{\tilde{\varphi}_y\tilde{y}} = \begin{bmatrix} r_{\tilde{y}}(1) & \cdots & r_{\tilde{y}}(n_a) \end{bmatrix}^T. \tag{2.60}$$

The $\lambda_{\min}$-equation, which is used here for the determination of $\sigma_{\tilde{u}}$, becomes for the coloured output noise case (cf. (2.46))

$$\sigma_{\tilde{u}} = \lambda_{\min}\left(\Sigma_{\varphi_u} - \Sigma_{\varphi_u\bar{\varphi}_y}\left[\Sigma_{\bar{\varphi}_y} - \Sigma_{\tilde{\bar{\varphi}}_y}\right]^{-1}\Sigma_{\varphi_y\varphi_u}\right). \tag{2.61}$$

Hence, it is necessary to replace Assumption **AN1** with the following.

AN1a The sequence $\tilde{u}_k$ is a zero-mean, ergodic, white noise process with unknown variance $\sigma_{\tilde{u}}$.

AN1b The sequence $\tilde{y}_k$ is a zero-mean, ergodic noise process with unknown auto-covariance sequence $\{r_{\tilde{y}}(0), r_{\tilde{y}}(1), \cdots\}$.

In the case of coloured output noise, the modified objective can thus be formulated as:

Problem 2.3. Given k samples of noisy input-output data $\{u_1, y_1, ..., u_k, y_k\}$, determine an estimate of the augmented parameter vector

$$\Theta \triangleq \begin{bmatrix} a_1 & ... & a_{n_a} & b_1 & ... & b_{n_b} & \rho_{\tilde{y}} & \sigma_{\tilde{u}} \end{bmatrix}^T \in \mathbb{R}^{2n_a+n_b+2}, \tag{2.62}$$

where $\rho_{\tilde{y}}$ is defined by (2.16).

Identification algorithm

In (Söderström 2006, Söderström 2008) several possibilities to obtain the remaining equations are discussed. It is shown that a covariance-matching criterion, as used in (Diversi, Guidorzi & Soverini 2003*a*), as well as correlating the residuals with past outputs, which corresponds to an instrumental variable-like approach with outputs as instruments, cannot be successful since it always leads to more unknowns than equations. However, by correlating the residuals, denoted ϵ_k, with past inputs, the remaining equations are obtained for the asymptotic case via

$$E\left[\zeta_k\epsilon_k\right] = 0, \tag{2.63}$$

where the instruments are given by (2.50) whilst the residuals are obtained via

$$\epsilon_k \triangleq A(q^{-1})y_k - B(q^{-1})u_k = y_k - \varphi_k^T\theta. \tag{2.64}$$

This yields

$$\xi_{\zeta y} - \Sigma_{\zeta\varphi}\theta = 0, \tag{2.65}$$

which can be expressed in block form as

$$\begin{bmatrix}\Sigma_{\zeta\varphi_y} & \Sigma_{\zeta\varphi_u}\end{bmatrix}\theta_k = \xi_{\zeta y}, \tag{2.66}$$

where the dimension n_ζ of the instrument vector ζ_k must satisfy $n_\zeta \geq n_a + 1$ in order to obtain at least $n_a + 1$ additional equations for the determination of $\rho_{\tilde{y}}$. For simplicity it is assumed that $n_\zeta = n_a + 1$ in the subsequent development. In summary, the $2n_a + nb + 2$ equations for the determination of the $2n_a + nb + 2$ unknowns are given as

$$\left(\begin{bmatrix}\Sigma_{\varphi_y} & \Sigma_{\varphi_y\varphi_u}\\ \Sigma_{\varphi_u\varphi_y} & \Sigma_{\varphi_u}\\ \Sigma_{\zeta\varphi_y} & \Sigma_{\zeta\varphi_u}\end{bmatrix} - \begin{bmatrix}\Sigma_{\tilde{\varphi}_y} & 0\\ 0 & \sigma_{\tilde{u}}I_{n_b}\\ 0 & 0\end{bmatrix}\right)\theta = \begin{bmatrix}\xi_{\varphi_y y}\\ \xi_{\varphi_u y}\\ \xi_{\zeta y}\end{bmatrix} - \begin{bmatrix}\xi_{\tilde{\varphi}_y\tilde{y}}\\ 0\\ 0\end{bmatrix}, \tag{2.67a}$$

$$\sigma_{\tilde{u}} = \lambda_{\min}\left(\Sigma_{\varphi_u} - \Sigma_{\varphi_u\bar{\varphi}_y}\left[\Sigma_{\bar{\varphi}_y} - \Sigma_{\tilde{\bar{\varphi}}_y}\right]^{-1}\Sigma_{\varphi_y\varphi_u}\right). \tag{2.67b}$$

In (Söderström 2006, Söderström 2008), two algorithms have been proposed to solve the resulting (nonlinear) estimation problem. Here, the two-step algorithm of (Söderström 2006), which makes use of the separable LS technique is considered. Whilst in the white noise case the estimate of θ is obtained from the compensated normal equations after the noise variances have been determined, the approach for coloured output noise is conceptually different as outlined in the remainder of this section.

Step 1 Note that $\rho_{\tilde{y}}$ only appears in the first n_a equations of (2.67a) and in (2.67b). By combining the last n_b equations of the compensated LS normal equations (2.58) with the $n_a + 1$ IV equations (2.66), one can formulate

$$\begin{bmatrix}\Sigma_{\varphi_u\varphi_y} & \Sigma_{\varphi_u} - \sigma_{\tilde{u}}I_{n_b}\\ \Sigma_{\zeta\varphi_y} & \Sigma_{\zeta\varphi_u}\end{bmatrix}\theta = \begin{bmatrix}\xi_{\varphi_u y}\\ \xi_{\zeta y}\end{bmatrix} \tag{2.68}$$

which constitute $n_a + n_b + 1$ equations in $n_a + n_b + 1$ unknowns (θ and $\sigma_{\tilde{u}}$). Equation (2.68) is an overdetermined system of normal equations with its first part obtained from the bias compensated LS and the second part given by the IV estimator, which uses delayed inputs. Moreover, it is nonlinear due to the multiplication of θ with $\sigma_{\tilde{u}}$.

In order to estimate θ and $\sigma_{\tilde{u}}$, (2.68) can be re-expressed as

$$(\Sigma_{\delta\varphi} - \sigma_{\tilde{u}} J)\,\theta = \xi_{\delta y}, \tag{2.69}$$

where $\Sigma_{\delta\varphi}$ and $\xi_{\delta y}$ are defined by (2.11) and (2.12) with

$$\delta_k \triangleq \begin{bmatrix} \varphi_{u_k}^T & \zeta_k^T \end{bmatrix}^T = \begin{bmatrix} u_{k-1} & \dots & u_{k-n_b} & u_{k-n_b-1} & \dots & u_{k-n_b-n_\zeta} \end{bmatrix}^T, \tag{2.70}$$

whilst J is given by

$$J \triangleq \begin{bmatrix} 0 & I_{n_b} \\ 0 & 0 \end{bmatrix}. \tag{2.71}$$

Note that (2.69) can be interpreted as a bias-compensated IV approach, where the instrument vector δ_k is constructed from past measured inputs. Using sample covariance estimates and introducing for convenience

$$G_k \triangleq \hat{\Sigma}_{\delta\varphi}^k - \sigma_{\tilde{u}} J, \tag{2.72}$$

the estimates for $\sigma_{\tilde{u}}$ and θ are obtained by minimising the (nonlinear) LS cost function

$$\{\hat{\theta}_k, \hat{\sigma}_{\tilde{u}}^k\} = \arg\min_{\theta,\sigma_{\tilde{u}}} \left|\left| G_k\theta - \xi_{\delta y}^k \right|\right|^2. \tag{2.73}$$

If $\sigma_{\tilde{u}}$ is assumed to be fixed, an explicit expression for $\hat{\theta}_k$ is given by the well-known LS solution

$$\hat{\theta}_k = G_k^\dagger \hat{\xi}_{\delta y}^k, \tag{2.74}$$

where $G_k^\dagger \triangleq [G_k^T G_k]^{-1} G_k^T$ denotes the Moore-Penrose pseudo inverse. Using the separable LS approach (see Section 2.3.2 or (Ljung 1999, p. 335)), the problem is reduced to an optimisation in one variable only by substituting (2.74) in (2.73). Consequently, $\hat{\sigma}_{\tilde{u}}^k$ can be obtained via

$$\hat{\sigma}_{\tilde{u}}^k = \arg\min_{\sigma_{\tilde{u}}} V_1^k \tag{2.75}$$

with

$$V_1^k = \left|\left| G_k G_k^\dagger \hat{\xi}_{\delta y}^k - \hat{\xi}_{\delta y}^k \right|\right|^2 = \hat{\xi}_{\delta y}^{kT} \hat{\xi}_{\delta y}^k - \hat{\xi}_{\delta y}^{kT} G_k \left[G_k^T G_k\right]^{-1} G_k^T \hat{\xi}_{\delta y}^k. \tag{2.76}$$

Once $\hat{\sigma}_{\tilde{u}}^k$ is obtained, $\hat{\theta}_k$ is then given by (2.74). Since the solution of (2.75) should satisfy $V_1^k = 0$, the value of V_1^k indicates whether the optimisation algorithm has converged to a global or local minimum (Söderström 2008).

Step 2 In order to determine the estimates for the n_a+1 terms of the auto-correlation sequence $\rho_{\tilde{y}}$, the remaining n_a normal equations (corresponding to the first n_a equations in (2.67a))

$$\left[\Sigma_{\varphi_y} - \Sigma_{\tilde{\varphi}_y}, \quad \Sigma_{\varphi_y\varphi_u}\right]\theta = \xi_{\varphi_y y} - \xi_{\tilde{\varphi}_y\tilde{y}} \tag{2.77}$$

together with the $\lambda_{\min}$-equation

$$\sigma_{\tilde{u}} = \lambda_{\min}\left(\Sigma_{\varphi_u} - \Sigma_{\varphi_u\bar{\varphi}_y}\left[\Sigma_{\bar{\varphi}_y} - \Sigma_{\tilde{\bar{\varphi}}_y}\right]^{-1}\Sigma_{\varphi_y\varphi_u}\right) \tag{2.78}$$

are considered. Note that for the computation of the compensation terms $\Sigma_{\tilde{\bar{\varphi}}_y}$ and $\xi_{\tilde{\varphi}_y\tilde{y}}$, the auto-covariance elements up to the time shift n_a are required, i.e. it is necessary to determine $\rho_{\tilde{y}}$. Using covariance estimates and replacing θ with its estimate $\hat{\theta}_k$ obtained in Step 1, Equation (2.77) can be rearranged and expressed as

$$\hat{\Sigma}^k_{\tilde{\varphi}_y}\hat{a}_k - \hat{\xi}^k_{\tilde{\varphi}_y\tilde{y}} = \left[\hat{\Sigma}^k_{\varphi_y} \quad \hat{\Sigma}^k_{\varphi_y\varphi_u}\right]\hat{\theta}_k - \hat{\xi}^k_{\varphi_y y}, \tag{2.79}$$

where only the left hand side depends on the unknown $\rho_{\tilde{y}}$. In addition, (2.79) is affine in $\rho_{\tilde{y}}$, hence it can be re-expressed as

$$H_k\rho_{\tilde{y}} = h_k, \tag{2.80}$$

where H_k is a $n_a \times n_a + 1$ matrix built up from elements of $\hat{a}_k$ and h_k is a vector of dimension n_a given by the right hand side of (2.79). This is a system of equations with more unknowns than equations, but the set of all possible solutions can be formulated as

$$\hat{\rho}^k_{\tilde{y}} = \alpha_k N(H_k) + H_k^{\dagger}h_k, \tag{2.81}$$

where $N(\cdot)$ denotes the nullspace and α_k is a scalar factor to be determined. It is necessary to distinguish between the input measurement noise variance obtained by (2.75) in Step 1, and the quantity which would be obtained by the $\lambda_{\min}$-equation (2.78). Therefore, introduce

$$\hat{\varsigma}_k \triangleq \lambda_{\min}\left(B_k\left(\alpha_k\right)\right), \tag{2.82}$$

where

$$B_k(\alpha_k) \triangleq \Sigma_{\varphi_u} - \Sigma_{\varphi_u\bar{\varphi}_y}\left[\Sigma_{\bar{\varphi}_y} - \Sigma_{\tilde{\bar{\varphi}}_y}(\alpha_k)\right]^{-1}\Sigma_{\varphi_y\varphi_u}. \tag{2.83}$$

Using (2.82) it is possible to search for that α_k which is in best agreement with the previously determined $\hat{\sigma}_{\tilde{u}}^k$, i.e.

$$\hat{\alpha}_k = \arg \min_{\alpha_k} V_2^k, \tag{2.84}$$

where

$$V_2^k = \left(\hat{\sigma}_{\tilde{u}}^k - \hat{\varsigma}_k\right)^2. \tag{2.85}$$

This means that the distance between the input noise variance estimate $\hat{\sigma}_{\tilde{u}}^k$ determined in Step 1 and the input noise variance estimate $\hat{\varsigma}_k$, which is obtained using the n_a normal equations (2.77) together with the λ_{min}-equation (2.82) depending on the choice of α_k, is minimised. Once $\hat{\alpha}_k$ is determined, it is substituted in (2.81) to obtain $\hat{\rho}_{\tilde{y}}^k$, the searched estimate of the auto-covariance elements of the coloured output measurement noise $\tilde{y}_k$.

Remark 2.1. Note that if only the determination of θ is of interest, Step 2 of the algorithm described above becomes dispensable. However, since one targeted application of the algorithms described within thesis is that of fault detection, the additional knowledge of the auto-covariance sequence might be beneficial and its estimation is therefore considered in the subsequent development.

Algorithm summary

The Frisch scheme for coloured output noise (FSCON) can be summarised as follows.

Algorithm 2.1 (FSCON).

Step1 Determine $\sigma_{\tilde{u}}$ and θ

$$\hat{\sigma}_{\tilde{u}}^k = \arg \min_{\hat{\sigma}_{\tilde{u}}^k} V_1^k \tag{2.86a}$$

$$\hat{\theta}_k = G_k^\dagger \hat{\xi}_{\delta y}^k(\hat{\sigma}_{\tilde{u}}^k) \tag{2.86b}$$

Step 2 Determine $\rho_{\tilde{y}}$

$$\hat{\alpha}_k = \arg \min_{\alpha_k} V_2^k \tag{2.86c}$$

$$\hat{\rho}_{\tilde{y}}^k = \alpha_k N(H_k) + H_k^\dagger h_k \tag{2.86d}$$

Remark 2.2. Note that Step 2 can be simplified significantly by acknowledging that

(2.77) and (2.78) can be jointly expressed as

$$\begin{bmatrix} \Sigma_{\bar{\varphi}_y} - \Sigma_{\tilde{\bar{\varphi}}_y} & \Sigma_{\bar{\varphi}_y \varphi_u} \end{bmatrix} \bar{\theta} = 0. \tag{2.87}$$

This allows the joint expression of (2.77) and (2.78) as

$$\bar{H}_k \rho_{\tilde{y}} = \bar{h}_k, \tag{2.88}$$

which constitutes $n_a + 1$ equations in $n_a + 1$ unknowns and can be solved in a straightforward manner (see Söderström 2008).

2.4.5 Joint output method

The joint output method (Söderström 1981) re-expresses the SISO EIV model as a multivariate state space system, which is driven by three white noise sequences. Introduce the following assumptions.

AI5 The noise-free system input u_{0_k} has a rational spectrum, i.e. it can be described as an ARMA process of the form

$$D(q^{-1}) u_{0_k} = C(q^{-1}) f_k, \tag{2.89}$$

where f_k is a white noise zero mean random process and the polynomials $C(q^{-1})$ and $D(q^{-1})$ are defined, respectively, by

$$C(q^{-1}) \triangleq 1 + c_1 q^{-1} + \cdots + c_{n_c} q^{-n_c}, \tag{2.90a}$$
$$D(q^{-1}) \triangleq 1 + d_1 q^{-1} + \cdots + d_{n_d} q^{-n_d}. \tag{2.90b}$$

AS5 The polynomials $A(q^{-1})$ and $B(q^{-1})$ are of the same order, i.e. $n \triangleq n_a = n_b$, whilst the polynomials $C(q^{-1})$ and $D(q^{-1})$ are chosen such that $n_c = n_d - 1$.

Assumption **AI5** is essential for the application of the joint output method, whereas Assumption **AS5** is introduced here for convenience only. Note that in the case of the joint output method, the parameter vector θ comprises the coefficients of $C(q^{-1})$ and $D(q^{-1})$ as well, i.e.

$$\theta = \begin{bmatrix} a_1 & \cdots & a_n & b_1 & \cdots & b_n & c_1 & \cdots & c_{n_c} & d_1 & \cdots & d_{n_d} \end{bmatrix}^T. \tag{2.91}$$

Consequently, the overall EIV system can be described as a state space system with a two dimensional output vector. One possibility is (Söderström 1981)

$$x_{k+1} = \mathscr{A}(\theta)x_k + \mathscr{B}(\theta)f_k, \tag{2.92a}$$

$$\begin{bmatrix} y_k \\ u_k \end{bmatrix} = \mathscr{C}(\theta)x_k + \begin{bmatrix} \tilde{y}_k \\ \tilde{u}_k \end{bmatrix}, \tag{2.92b}$$

where $x_k \in \mathbb{R}^{n_a+n_b}$ denotes the system state vector and where

$$\mathscr{A}(\theta) \triangleq \begin{bmatrix} \mathscr{A}_{11}(\theta) & \mathscr{A}_{12}(\theta) \\ \mathscr{A}_{21}(\theta) & \mathscr{A}_{22}(\theta) \end{bmatrix} \in \mathbb{R}^{(n+n_d)\times(n+n_d)}, \tag{2.93a}$$

$$\mathscr{B}(\theta) \triangleq \begin{bmatrix} \mathscr{B}_1(\theta) \\ \mathscr{B}_2(\theta) \end{bmatrix} \in \mathbb{R}^{n+n_d} \tag{2.93b}$$

$$\mathscr{C}(\theta) \triangleq \begin{bmatrix} 1 & 0_{1\times(n-1)} & 0_{1\times n_d} \\ 0_{1\times n} & 1 & 0_{1\times n_d} \end{bmatrix} \in \mathbb{R}^{2\times(n+n_d)} \tag{2.93c}$$

with

$$\mathscr{A}_{11}(\theta) \triangleq \begin{bmatrix} -a_1 & 1 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ \vdots & 0 & 0 & 1 \\ -a_n & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{n\times n}, \quad \mathscr{A}_{12}(\theta) \triangleq \begin{bmatrix} b_1 & 0 \\ \vdots & 0 \\ b_n & 0 \end{bmatrix}, \in \mathbb{R}^{n\times n_d} \tag{2.94}$$

$$\mathscr{A}_{21}(\theta) \triangleq 0_{n_d\times n} \qquad \mathscr{A}_{22}(\theta) \triangleq \begin{bmatrix} -d_1 & 1 & 0 & 0 \\ \vdots & 0 & \ddots & 0, \\ \vdots & 0 & 0 & 1 \\ -d_{n_d} & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{n_d\times n_d},$$

and

$$\mathscr{B}_1(\theta) \triangleq 0_{n\times 1}, \qquad \mathscr{B}_2(\theta) \triangleq \begin{bmatrix} 1 & c_1 & \cdots & c_{n_c} \end{bmatrix}^T \in \mathbb{R}^{n_d}. \tag{2.95}$$

Note that the first element of x_k corresponds to y_{0_k}, whilst the $(n+1)$th element of x_k corresponds to u_{0_k}. By applying the Kalman filter (see below in Section 2.5.1) it is possible to obtain the optimal state estimate, hence estimates of the noise-free input and noise-free output, which are denoted $\hat{u}_{0_k}$ and $\hat{y}_{0_k}$, respectively. Based on this filtered input and output, it is possible to design a two dimensional innovations vector

$$\epsilon_k(\theta) \triangleq \begin{bmatrix} y_k - \hat{y}_{0_k} \\ u_k - \hat{u}_{0_k} \end{bmatrix}, \tag{2.96}$$

which is termed the symmetric innovation within this thesis. It is then possible to define the cost function

$$V_\epsilon(\theta) = \frac{1}{2} E\left[\epsilon_k^T(\theta)\epsilon_k(\theta)\right], \tag{2.97}$$

which leads to the well-known prediction error method (Ljung 1999). In addition, it is possible to obtain the maximum likelihood estimate in the case of Gaussian data (see Söderström 2007*b*). The minimisation of (2.97) is, however, rather computationally demanding, since a Riccati equation is required to be solved at each iteration step, in order to obtain the symmetric innovations $\{\epsilon_k(\theta)\}_{k=1}^N$, where N denotes the number of available samples.

Note that the algorithm can also be formulated using frequency domain data (see Pintelon & Schoukens 2007).

2.5 Filtering

Filtering can be defined as the problem of determining the unknown state of a dynamical system from noise corrupted measurements (Jazwinski 1970). It is therefore an estimation problem which has, under certain assumptions, an optimal solution in the linear Gaussian case. This is the well known Kalman filter (KF) which has been proposed by R. E. Kalman in 1960 (Kalman 1960). This section first reviews the KF algorithm as well as some modifications, namely the errors-in-variables KF (EIVKF) and the extended KF for joint state and parameter estimation (JEKF), which form the bases for further developments within this thesis.

2.5.1 Kalman filtering

Consider the discrete-time linear dynamic state space system given, for $k \geq 0$, by

$$x_{k+1} = A_k x_k + B_k u_k + G_k v_k, \qquad x_0 = \bar{x}_0, , \tag{2.98a}$$

$$z_k = C_k x_k + D_k u_k + e_k, \tag{2.98b}$$

where $x_k \in \mathbb{R}^{n_x}$ denotes the system state vector, $\bar{x}_0$ denotes the mean of the initial state vector x_0 and $A_k \in \mathbb{R}^{n_x \times n_x}$, $B_k \in \mathbb{R}^{n_x \times n_u}$, $C_k \in \mathbb{R}^{n_z \times n_x}$, $D_k \in \mathbb{R}^{n_u \times n_u}$ and $G_k \in \mathbb{R}^{n_x \times 1}$ are matrices of appropriate dimension[6]. The noise sequences $v_k \in \mathbb{R}$ and $e_k \in \mathbb{R}^{n_z}$ denote process and measurement noise, respectively. The following assumptions are introduced.

AI2 The system input u_k is known exactly.

[6]Note that in this non-EIV situation the input $u_k \in \mathbb{R}^{n_u}$ is assumed to be known exactly, whilst the measured output is denoted by $z_k \in \mathbb{R}^{n_z}$.

AN3 The noise sequences v_k and e_k are zero mean, white, and satisfy

$$E\left[\begin{bmatrix} v_k \\ e_k \end{bmatrix}\begin{bmatrix} v_l^T & e_l^T \end{bmatrix}\right] = \begin{bmatrix} \Sigma_v^k & \Sigma_{ve}^k \\ \Sigma_{ve}^{k^T} & \Sigma_e^k \end{bmatrix}\delta_{kl}. \tag{2.99}$$

AN4 The initial state x_0 has the mean $\bar{x}_0$ with covariance matrix P_0. In addition, x_0 is independent of $\begin{bmatrix} v_k^T & e_k^T \end{bmatrix}^T$ for all k.

AN5 The quantities x_0, v_k and e_k are jointly Gaussian.

The optimal one-step ahead prediction of the state, denoted $\hat{x}_{k+1|k}$, is given by the well known KF, which has been developed in (Kalman 1960). Here, optimal refers to the unbiased minimum variance property of the estimates. The single phase form of the KF equations are given by the following algorithm (Anderson & Moore 1979, Sec. 5.4).

Algorithm 2.2 (Single phase KF).

$$\hat{x}_{k+1|k} = A_k\hat{x}_{k|k-1} + B_k u_k + K_k\left[z_k - C_k\hat{x}_{k|k-1} - D_k u_k\right] \tag{2.100a}$$

$$K_k = \left[A_k P_{k|k-1}C_k^T + G_k\Sigma_{ve}^k\right]\left[C_k P_{k|k-1}C_k^T + \Sigma_e^k\right]^{-1} \tag{2.100b}$$

$$P_{k+1|k} = A_k P_{k|k-1}A_k^T + G_k\Sigma_v^k G_k^T - K_k\left[C_k P_{k|k-1}C_k^T + \Sigma_e^k\right]K_k^T \tag{2.100c}$$

The vector K_k denotes the Kalman gain and $P_{k+1|k}$ denotes the error covariance matrix of the estimated states. A thorough treatment of Kalman filtering theory can be found in (Anderson & Moore 1979, Kailath & Sayed 2000), whilst a historical survey is given in (Sorenson 1970). The relation between RLS and the KF is discussed in (Sayed & Kailath 1994*a*, Young 1974, Young 1984).

2.5.2 Errors-in-variables filtering

In the non-EIV case, Kalman filtering deals with the optimal estimation of states and outputs in the presence of process and output noise. EIV filtering, in contrast, aims at estimating the noise-free outputs and noise-free inputs in the case where both quantities are corrupted by additive noise. This problem was first addressed in (Guidorzi et al. 2003), where a residual model representation of the EIV system (2.1)-(2.3) has been considered. By formulating a state space model representation for the residuals, an optimal (in the unbiased minimum variance sense) estimator for the noise-free inputs u_{0_k} and outputs y_{0_k} has been derived; different implementations have also been discussed in (Diversi et al. 2003*a*).

An alternative formulation leading to an identical solution has been reported in (Markovsky & De Moor 2005), where the optimal filter is directly derived from an

EIV state space representation by making use of the well known KF algorithm. Whilst both approaches yield equivalent results, the derivation in (Markovsky & De Moor 2005) appears to be more straightforward. This is because the problem is essentially solved by interpreting the input measurement noise as process noise (with accordingly modified covariance matrices), which allows the KF to be applied directly. Both approaches consider a zero residual problem (Van Huffel & Vandewalle 1991), that is, the noise-free input and output signals are related by an exact linear relationship, i.e. no equation error or process noise is considered. Following the approach in (Markovsky & De Moor 2005), a unified framework for EIV and Kalman filtering has been presented in (Diversi et al. 2005), where measurement noise as well as process disturbances are considered. The relationship between EIV filtering and unknown input Kalman filtering is discussed in (Gillijns & De Moor 2006), where the derived EIV filter allows a linear combination of the input vector to be observed instead of the entire input vector.

The errors-in-variables Kalman filter (EIVKF) as presented in (Diversi et al. 2005) is reviewed in the following.

Errors-in-variables Kalman filter

In (Diversi et al. 2005), a general EIV state space model

$$x_{k+1} = \mathcal{A}_k x_k + \mathcal{B}_k u_{0_k} + \mathcal{G}_k w_k, \quad x_0 = \bar{x}_0, \tag{2.101a}$$

$$y_{0_k} = \mathcal{C}_k x_k + \mathcal{D}_k u_{0_k}, \tag{2.101b}$$

$$u_k = u_{0_k} + \tilde{u}_k, \tag{2.101c}$$

$$y_k = y_{0_k} + \tilde{y}_k, \tag{2.101d}$$

is considered, which allows for input and output additive measurement noise, denoted by $\tilde{u}_k$ and $\tilde{y}_k$, respectively, as well as process noise, denoted by w_k. Here, $x_k \in \mathbb{R}^{n_x}$ denotes the state of the system, and $\mathcal{A}_k$, $\mathcal{B}_k$, $\mathcal{C}_k$, $\mathcal{D}_k$ and $\mathcal{G}_k$ are matrices of appropriate dimension. The initial state x_0 is a random vector with mean $\bar{x}_0$ and covariance matrix P_0. In contrast to the non-EIV state space representation given in (2.98), the true input $u_{0_k} \in \mathbb{R}^{n_u}$ of the system is unknown and only the measured quantity u_k is available, which is affected by additive measurement noise $\tilde{u}_k$. The following assumptions are introduced.

AN6a The noise sequences $\tilde{u}_k$, $\tilde{y}_k$ and w_k are assumed to be zero mean, white, independent of u_{0_k} and are characterised by the known covariance matrices

$$E\left[\begin{bmatrix} x_0 \\ \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix} \begin{bmatrix} x_0^T & \tilde{u}_l^T & \tilde{y}_l^T & w_l \end{bmatrix}\right] = \begin{bmatrix} P_0 & 0 & 0 & 0 \\ 0 & \Sigma_{\tilde{u}}^k & \Sigma_{\tilde{u}\tilde{y}}^k & 0 \\ 0 & \Sigma_{\tilde{u}\tilde{y}}^{k^T} & \Sigma_{\tilde{y}}^k & 0 \\ 0 & 0 & 0 & \Sigma_w^k \end{bmatrix} \delta_{kl}. \tag{2.102}$$

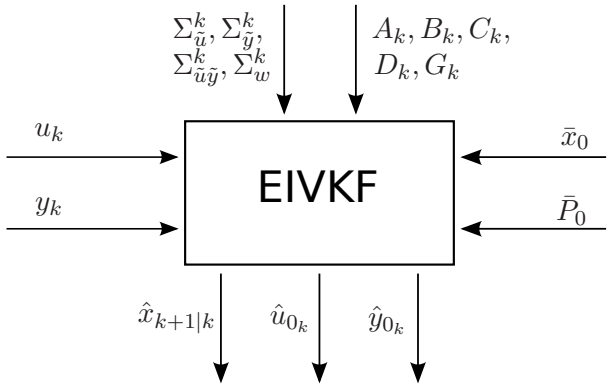


Figure 2.3: Inputs and outputs of the EIVKF.

As before, $E[\cdot]$ denotes the expected value operator and δ_{kl} the Kronecker delta function. Note that the input and output noise sequences are allowed to be correlated in this setup. Indeed, the correlation between both sequences plays a crucial role in order to obtain an estimate of u_{0_k} (cf. Remark 2.3 below). Figure 2.3 shows a block diagram of the EIVKF with corresponding inputs (known quantities) and outputs (estimated quantities).

The system representation (2.101) can be reformulated by introducing the transformations

$$v_k = \mathcal{G}_k w_k - \mathcal{B}_k \tilde{u}_k, \tag{2.103a}$$
$$z_k = y_k - \mathcal{D}_k u_k, \tag{2.103b}$$
$$e_k = \tilde{y}_k - \mathcal{D}_k \tilde{u}_k, \tag{2.103c}$$

which gives the equivalent state-space representation

$$x_{k+1} = \mathcal{A}_k x_k + \mathcal{B}_k u_k + v_k, \tag{2.104a}$$
$$z_k = \mathcal{C}_k x_k + e_k, \tag{2.104b}$$

with corresponding covariance matrices

$$\Sigma_v = \mathcal{G}_k \Sigma_w \mathcal{G}_k^T + \mathcal{B}_k \Sigma_{\tilde{u}} \mathcal{B}_k^T, \tag{2.105a}$$
$$\Sigma_e = \Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T \mathcal{D}_k^T - \mathcal{D}_k \Sigma_{\tilde{u}\tilde{y}} + \mathcal{D}_k \Sigma_{\tilde{u}} \mathcal{D}_k^T, \tag{2.105b}$$
$$\Sigma_{ve} = \mathcal{B}_k \left[\Sigma_{\tilde{u}} \mathcal{D}_k^T - \Sigma_{\tilde{u}\tilde{y}} \right]. \tag{2.105c}$$

Consequently, a standard single phase KF (one-step prediction state estimator, (see Chapter 5.4 in Anderson & Moore 1979)) can be applied, which is given by the following algorithm.

Algorithm 2.3 (EIVKF).

$$\hat{x}_{k+1|k} = \mathcal{A}_k \hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k \varepsilon_k \tag{2.106a}$$
$$\varepsilon_k = z_k - \mathcal{C}_k \hat{x}_{k|k-1} \tag{2.106b}$$
$$\Sigma_\varepsilon^k = \mathcal{C}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_e^k \tag{2.106c}$$
$$K_k = \left[\mathcal{A}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_{ve}^k\right] \left[\Sigma_\varepsilon^k\right]^{-1} \tag{2.106d}$$
$$P_{k+1|k} = \mathcal{A}_k P_{k|k-1} \mathcal{A}_k^T + \Sigma_v^k - K_k \Sigma_\varepsilon^k K_k^T \tag{2.106e}$$
$$\hat{y}_{0_k} = y_k - \left[\Sigma_{\tilde{y}}^k - \Sigma_{\tilde{u}\tilde{y}}^{k^T} \mathcal{D}_k^T\right] \left[\Sigma_\varepsilon^k\right]^{-1} \varepsilon_k \tag{2.106f}$$
$$\hat{u}_{0_k} = u_k - \left[\Sigma_{\tilde{u}\tilde{y}}^k - \Sigma_{\tilde{u}}^k \mathcal{D}_k^T\right] \left[\Sigma_\varepsilon^k\right]^{-1} \varepsilon_k \tag{2.106g}$$

Note that (2.106a)-(2.106e) is identical to Algorithm 2.2 where $D_k = 0$ and $G_k = I$, with ε_k and Σ_ε^k being the innovations and corresponding covariance matrix, respectively. In fact, the only difference with respect to standard Kalman filtering for state estimation is the symmetric computation of the filtered inputs and outputs, which are obtained via (Diversi et al. 2005)

$$\hat{u}_{0_k} = u_k - E\left[\tilde{u}_k | z_k\right], \tag{2.107a}$$
$$\hat{y}_{0_k} = y_k - E\left[\tilde{y}_k | z_k\right]. \tag{2.107b}$$

The expected filter performance can be evaluated via the error covariance matrices

$$\begin{aligned} P_u^k &= E\left[\left[u_{0_k} - \hat{u}_{0_k}\right]\left[u_{0_k} - \hat{u}_{0_k}\right]^T\right] \\ &= \Sigma_{\tilde{u}}^k - \left[\Sigma_{\tilde{u}\tilde{y}}^k - \Sigma_{\tilde{u}}^k \mathcal{D}_k^T\right] \left[\Sigma_\varepsilon^k\right]^{-1} \left[\Sigma_{\tilde{u}\tilde{y}}^k - \Sigma_{\tilde{u}}^k \mathcal{D}_k^T\right]^T, \end{aligned} \tag{2.108a}$$
$$\begin{aligned} P_y^k &= E\left[\left[y_{0_k} - \hat{y}_{0_k}\right]\left[y_{0_k} - \hat{y}_{0_k}\right]^T\right] \\ &= \Sigma_{\tilde{y}}^k - \left[\Sigma_{\tilde{y}}^k - \Sigma_{\tilde{u}\tilde{y}}^{k^T} \mathcal{D}_k^T\right] \left[\Sigma_\varepsilon^k\right]^{-1} \left[\Sigma_{\tilde{y}}^k - \Sigma_{\tilde{u}\tilde{y}}^{k^T} \mathcal{D}_k^T\right]^T. \end{aligned} \tag{2.108b}$$

Remark 2.3. The estimation of the noise-free input is only possible if either $\mathcal{D} \neq 0$ and/or the input and output measurement noise is correlated. If $\mathcal{D} = 0$ and $\Sigma_{\tilde{u}\tilde{y}}^k = 0$ the estimate in (2.106g) becomes u_k and the corresponding estimation error covariance matrix in (2.108a) is identical to $\Sigma_{\tilde{u}}^k$.

2.5.3 Kalman filtering for bilinear systems

Whilst EIV filtering has only been considered for linear systems within the literature, a part of this thesis addresses the EIV filtering problem for bilinear systems, a particular

class of nonlinear systems. These are reviewed in the following together with the corresponding KF.

Bilinear systems

Bilinear model representations are an appealing class of nonlinear models, since they are considered to be 'nearly linear' due to their close structural and behavioural connections with linear models (Pearson 1999). As bilinear characteristics arise in numerous areas such as engineering, socioeconomics, chemistry and biology (Mohler 1991) and are able to approximate many dynamical processes, there is a natural interest and motivation to study such models. In the discrete-time domain, there exist at least two different definitions of bilinear models: state-space and input-output representations. In contrast to their linear counterparts, however, these two representations are in general not equivalent (Pearson 1999, Pearson & Kotta 2004) and an investigation of the relationship between these two bilinear model classes, together with some general results, is presented in (Pearson & Kotta 2004). Furthermore, the aspect of stochastic bilinear realisation theory, i.e. the task of finding bilinear state-space realisations from input-output representations, is addressed in (Favoreel, De Moor & Van Overschee 1999). The recursive identification of discrete-time bilinear input-output models is considered in (Fnaiech & Ljung 1987) whereas a subspace approach for the identification of bilinear state-space models is considered in (Favoreel et al. 1999, Verdult 2002). An approach for the identification of a class of bilinear EIV models by means of extended compensated least-squares is presented in (Ekman 2005).

Kalman filter for bilinear systems

Consider the bilinear time-invariant single-input single-output (SISO) discrete-time (non-EIV) state-space representation, which is given by

$$x_{k+1} = Ax_k + Bu_k + Nu_kx_k + Gv_k, \qquad x_0 = \bar{x}_0, \tag{2.109a}$$

$$z_k = Cx_k + Du_k + e_k, \tag{2.109b}$$

where $x_k \in \mathbb{R}^{n_x}$ is the state vector, x_0 its initial value with mean $\bar{x}_0$ and covariance matrix P_0, whilst $u_k \in \mathbb{R}$ and $z_k \in \mathbb{R}$ are the system input[7] and measured output, respectively. The quantity v_k is the noise acting on the state and e_k denotes the measurement output noise. The matrices A, B, C, D, G and N are known, time-invariant and of appropriate dimensions. The difference between the bilinear and the linear case is the additional multiplicative term involving the state and input.

Remark 2.4 (Multiple-input multiple-output case). The restriction to the SISO case is considered here without loss of generality and for convenience only. In the multivariate

[7]The system input is assumed to be known for the time-being.

case, i.e. $u_k \in \mathbb{R}^{n_u}$ and $y_k \in \mathbb{R}^{n_y}$, the bilinear term in (2.109a) becomes $Nu_k \otimes x_k$, where the Kronecker product $\otimes$ is defined such that $c \otimes d = [c_1 d^T \quad \cdots \quad c_p d^T]^T \in \mathbb{R}^{pq}$ with $c \in \mathbb{R}^p$ and $d \in \mathbb{R}^q$ being arbitrary vectors. In addition, $N = [N_1 \quad \cdots \quad N_{n_u}] \in \mathbb{R}^{n_x \times n_x n_u}$ holds, which allows linear combinations of multiplicative terms between each input and each element of the state vector.

It is observed, that the bilinear model (2.109) is linear in the input and linear in the states, but jointly nonlinear in state and input. The fact that bilinear models are linear in the states allows (2.109) to be considered as a linear time-varying model (Fnaiech & Ljung 1987, Favoreel et al. 1999). This property is highlighted by factorising the state or the input term, from which it is clear that this either leads to a time-varying (input-dependent) system matrix

$$A_k \triangleq A + Nu_k, \tag{2.110}$$

or a time-varying (state-dependent) input matrix

$$B_k \triangleq B + Nx_k, \tag{2.111}$$

respectively. Hence, a KF for state estimation of (the non-EIV) model (2.109) can be derived in a straightforward manner as for linear state-space models (cf. Favoreel et al. 1999, Ekman 2005) using the time-varying system matrix A_k. Note that for the case of jointly Gaussian state noise v_k and initial state $\bar{x}_0$, where the latter is independent of v_k and $\tilde{y}_k$ for all k and whose first and second order statistics are known, x_k is Gaussian too, since (2.109a) is linear in the state. Hence, in the case of known input u_k, the KF for the bilinear model (2.109) will be the optimal filter in a minimum mean-square error sense (Anderson & Moore 1979). A derivation of the bilinear KF is given in Appendix H.

Following the interpretation of a bilinear system as a linear time-varying (input dependent) system as in (2.110), it is apparent that the input signal will influence the stability of the system. Therefore, it seems natural to introduce the following assumption.

AI4 The system input u_k behaves in a manner, such that the bilinear system whose dynamics are characterised by the state transition matrix $A + Nu_k$ is stable.

Note that in the time-varying case, it is neither necessary nor sufficient to demand that the spectral radius of $A + Nu_k$ is less than unity for all k (see e.g. Stoica & Söderström 1995).

Systems which can be described by (2.109) are commonly referred to as bilinear systems, whereas if the input u_k is considered to be a random signal rather than a deterministic sequence, (2.109) is known as a bilinear stochastic system (Carravetta, Germani & Raimondi 1997). Alternatively, a bilinear stochastic system can be viewed

as a system with multiplicative state noise or, equivalently, as a linear system with a random system matrix (De Koning 1984). This point of view can also be taken in the case of bilinear EIV systems, as outlined in Chapter 6.

2.5.4 Extended Kalman filter for joint state and parameter estimation

When the states of a linear state space model are extended with the parameter vector θ, an extended Kalman filter can be utilised to estimate the states as well as the system parameters in a joint fashion. This algorithm is the well known extended Kalman filter for joint state and parameter estimation (JEKF) (Ljung 1979), which is reviewed as follows.

Preliminaries

Consider the input-output data created by the (non-EIV) discrete-time LTI state space system given, for $k \geq 0$, by

$$x_{k+1} = A_0 x_k + B_0 u_k + v_k, \qquad x_0 = \bar{x}_0 \tag{2.112a}$$

$$z_k = C_0 x_k + D_0 u_k + e_k, \tag{2.112b}$$

where $x_k \in \mathbb{R}^{n_x}$ denotes the state, u_k the input, z_k the output, v_k the process noise, e_k the measurement noise and the 'true' system matrices A_0, B_0, C_0 and D_0 are of appropriate dimensions. As in Section 2.5.1, the assumptions **AI2** and **AN3-5** hold. Also introduce the corresponding model which is given by

$$x_{k+1} = A(\theta) x_k + B(\theta) u_k + v_k, \tag{2.113a}$$

$$z_k = C(\theta) x_k + D(\theta) u_k + e_k, \tag{2.113b}$$

where the matrices $A(\theta)$, $B(\theta)$, $C(\theta)$ and $D(\theta)$ are dependent on the parameter vector θ, which defines the system.

Standard form

Based on an extended Kalman filter (EKF) (Jazwinski 1970, Anderson & Moore 1979) an adaptive estimator for the model parameters can be derived by extending the state with the time dependent parameter vector, denoted θ_k, which leads to the following

nonlinear[8] state space representation

$$\begin{bmatrix} x_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} A(\theta_k)x_k + B(\theta_k)u_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} v_k \\ d_k \end{bmatrix}, \tag{2.114a}$$

$$z_k = C(\theta_k)x_k + D(\theta_k)u_k + e_k. \tag{2.114b}$$

The noise term d_k with covariance matrix

$$\Sigma_d \delta_{kl} = E\left[d_k d_l^T\right] \tag{2.115}$$

allows for variations in the system parameters and is usually set to zero if time-invariance is assumed.

Defining for convenience

$$\begin{aligned} A_k &\triangleq A(\hat{\theta}_k), & B_k &\triangleq B(\hat{\theta}_k), \\ C_k &\triangleq C(\hat{\theta}_k), & D_k &\triangleq D(\hat{\theta}_k), \end{aligned} \tag{2.116}$$

the EKF for joint state and parameter estimation (JEKF) is given by the following algorithm (Ljung 1979) (see also Appendix J for a detailed derivation).

Algorithm 2.4 (JEKF).

$$\hat{x}_{k+1|k} = A_k\hat{x}_{k|k-1} + B_k u_k + K_k\left[z_k - C_k\hat{x}_{k|k-1} - D_k u_k\right] \tag{2.117a}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + L_k\left[z_k - C_k\hat{x}_{k|k-1} - D_k u_k\right] \tag{2.117b}$$

$$K_k = [A_k P_{1_k} C_k^T + F_k P_{2_k}^T C_k^T + A_k P_{2_k} H_k^T + F_k P_{3_k} H_k^T + \Sigma_{ve}]S_k^{-1} \tag{2.117c}$$

$$S_k = C_k P_{1_k} C_k^T + C_k P_{2_k} H_k^T + H_k P_{2_k}^T C_k^T + H_k P_{3_k} H_k^T + \Sigma_e \tag{2.117d}$$

$$L_k = \left[P_{2_k}^T C_k^T + P_{3_k} H_k^T\right] S_k^{-1} \tag{2.117e}$$

$$P_{1_{k+1}} = A_k P_{1_k} A_k^T + A_k P_{2_k} F_k^T + F_k P_{2_k}^T A_K^T + F_k P_{3_k} F_k^T - K_k S_k K_k^T + \Sigma_v \tag{2.117f}$$

$$P_{2_{k+1}} = A_k P_{2_k} + F_k P_{3_k} - K_k S_k L_k^T \tag{2.117g}$$

$$P_{3_{k+1}} = P_{3_k} - L_k S_k L_k^T + \Sigma_d \tag{2.117h}$$

The Jacobians in Algorithm 2.4 are defined by

$$F_k = F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k), \tag{2.118a}$$

$$H_k = H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k), \tag{2.118b}$$

[8] Nonlinear due to the product between x_k and θ_k, where the latter is incorporated within $A(\theta_k)$ and $C(\theta_k)$. Since both, x_k and θ_k are part of the newly formed augmented state vector, the resulting equations are nonlinear.

where

$$F(\hat{\theta}, x, u) = \frac{\partial}{\partial \theta} \left[A(\theta)x + B(\theta)u \right] \big|_{\theta=\hat{\theta}}, \tag{2.119a}$$

$$H(\hat{\theta}, x, u) = \frac{\partial}{\partial \theta} \left[C(\theta)x + D(\theta)u \right] \big|_{\theta=\hat{\theta}}. \tag{2.119b}$$

Innovations form

It is shown in (Ljung 1979) that the above recursive parameter estimator can be interpreted as an attempt to minimise the expected value of squared prediction errors associated with a constant model θ. Hence, this estimator is closely related to a recursive prediction error method. In fact, the recursive prediction error method applied to a state space model and the parameter estimator derived by the EKF are virtually identical, except that the term corresponding to the derivative of the Kalman gain (which is a function of θ) has been discarded (Ljung & Söderström 1983, p. 130). A convergence analysis of this parameter estimator for linear systems is also carried out in (Ljung 1979) and it is shown that it can produce biased estimates or it can even diverge. However, the above procedure can be modified to become a stochastic descent-algorithm which is globally convergent by including an approximation of the dismissed term

$$\left[\frac{d}{d\theta} K(\theta) \right] \varepsilon_k \tag{2.120}$$

into the Jacobian F_k (referred to as the coupling term in (Ljung 1979)), where $K(\theta)$ is the Kalman gain and ε_k denotes the innovation defined by

$$\varepsilon_k \triangleq z_k - C(\theta)\hat{x}_{k|k-1} - D(\theta)u_k. \tag{2.121}$$

One way to ensure this property is to assume an innovation model structure

$$x_{k+1} = A(\theta)x_k + B(\theta)u_k + K(\theta)\varepsilon_k, \tag{2.122a}$$

$$z_k = C(\theta)x_k + D(\theta)u_k + \varepsilon_k, \tag{2.122b}$$

rather than (2.113) and include all elements of the Kalman gain K into the parameter vector θ, i.e.

$$\theta_k = \begin{bmatrix} a_1 & \cdots & a_{n_a} & b_1 & \cdots & b_{n_b} & k_1 & \cdots & k_{n_x} \end{bmatrix}^T. \tag{2.123}$$

It is argued in (Ljung 1979) that the noise covariance information is solely used to arrive at the Kalman gain (2.117c), and if the latter is estimated directly as part of the extended state vector, a more parsimonious parametrisation is achieved. Hence the innovation parametrisation is generally beneficial, if no explicit a priori information of the noise structure in the form of the covariance matrices is available. Consequently,

parameterising $K(\theta)$ and Σ_ε explicitly leads to a modified algorithm which can be summarised as follows.

Algorithm 2.5 (JEKF in innovation form).

$$\hat{x}_{k+1|k} = A_k\hat{x}_{k|k-1} + B_k u_k + K_k\varepsilon_k \tag{2.124a}$$
$$\hat{\theta}_{k+1} = \hat{\theta}_k + L_k\varepsilon_k \tag{2.124b}$$
$$\varepsilon_k = z_k - C_k\hat{x}_{k|k-1} - D_k u_k \tag{2.124c}$$
$$L_k = \left[P_{2_k}^T C_k^T + P_{3_k} H_k^T\right]\left[\Sigma_\varepsilon^k\right]^{-1} \tag{2.124d}$$
$$P_{2_{k+1}} = A_k P_{2_k} + \bar{F}_k P_{3_k} - K_k\Sigma_\varepsilon^k L_k^T \tag{2.124e}$$
$$P_{3_{k+1}} = P_{3_k} - L_k\Sigma_\varepsilon^k L_k^T + \Sigma_d \tag{2.124f}$$
$$\Sigma_\varepsilon^k = \Sigma_\varepsilon^{k-1} + \frac{1}{k}\left(\varepsilon_k\varepsilon_k^T - \Sigma_\varepsilon^{k-1}\right) \tag{2.124g}$$

The quantities $\bar{F}_k$ and K_k are given by

$$\bar{F}_k = \bar{F}(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k, \varepsilon_k), \tag{2.125a}$$
$$K_k = K(\hat{\theta}_k), \tag{2.125b}$$

with

$$\bar{F}(\hat{\theta}, x, u, \varepsilon) = \frac{\partial}{\partial\theta}\left[A(\theta)x + B(\theta)u + K(\theta)\varepsilon\right]\big|_{\theta=\hat{\theta}}, \tag{2.126}$$

whilst H_k is given by (2.118b). Note that the innovation covariance matrix is also estimated from the data via (2.124g). In addition, a projection facility has to be utilised to ensure that $\hat{\theta}_k$ lies in the compact subset defined by

$$D_s = \{\theta | A(\theta) - K(\theta)C(\theta) \textit{ is exponentially stable}\}, \tag{2.127}$$

which means that the poles of the KF are projected into the unit circle. In practice, a step-size reduction might also be necessary to achieve convergence.

2.6 Concluding remarks

This Chapter has reviewed a few well-known tools and techniques for the errors-in-variables identification and filtering problem, hence providing a detailed literature review of the subject. Whilst a more detailed disquisition of the reviewed methods can be found within the cited literature, this chapter has equipped the reader with the fundamental concepts, which allow the developments of the forthcoming chapters to be

followed.

Chapter 3

Gradient-based recursive Frisch scheme approaches

Contents

Nomenclature

$\hat{A}_k$	Schur complement
d_1^k, d_2^k, d_3^k, d_4^k, d_5^k	Auxiliary terms
F_1^k, F_2^k, F_3^k	Measures for Frisch-character
$J(\sigma_{\tilde{u}})$	Jacobian (of residual $r_k(\theta)$ with respect to $\sigma_{\tilde{u}}$)
$J_k^{(\sigma_{\tilde{u}})}$	Approximate Jacobian (of residual $r_k(\theta)$ with respect to $\sigma_{\tilde{u}}$)
L_k	Recursive least squares gain
$L_\theta(\vartheta)$	Linearised θ-equation of Frisch scheme
$L_{\sigma_{\tilde{y}}}(\vartheta)$	Linearised $\lambda_{\min}$-equation of Frisch scheme
MSE_k	Mean square error
P_k	Scaled covariance matrix obtained from Recursive least squares
$r_k(\theta)$	Residual of Yule-Walker cost function
$\hat{r}_k$	Residual of conjugate gradient method
$R(x_k)$	Rayleigh quotient
V_k	YW cost function
V_k', V_k''	First and second order derivative of V_k
V_k^{lin}	Yule-Walker cost function using linearised Frisch scheme equations
$V_k^{(\theta)}$	Approximate derivative of V_k with respect to θ
$V_k^{(\sigma_{\tilde{u}})}$	Approximate derivative of V_k with respect to $\sigma_{\tilde{u}}$
x_k	Eigenvector corresponding to $\hat{A}_k$
$\bar{x}_k$	Eigenvector x_k scaled to unity length
β_i^k	Weighting (of ith data at time k)
γ_k	Gain sequence or step size
ζ_k	Instrument vector comprising delayed inputs
$\theta_k^{(\tilde{y})}$	Approximate derivative of $\hat{\theta}_k$ with respect to $\hat{\sigma}_{\tilde{y}}^k$
$\theta_k^{(\tilde{u})}$	Approximate derivative of $\hat{\theta}_k$ with respect to $\hat{\sigma}_{\tilde{u}}^k$
$\hat{\theta}_{k+\frac{1}{2}}$	Intermediate estimate of θ
ϑ_*	Point of linearisation
$\iota(\vartheta)$	Auxiliary term
$\bar{\iota}(\vartheta)$	Auxiliary term
$\kappa(\vartheta)$	Auxiliary term
λ_k	Forgetting factor
$\hat{\mu}_k$	Step size (conjugate gradient)
$\sigma_{\tilde{y}}^{\max}$	Maximal admissible value for $\hat{\sigma}_{\tilde{y}}^k$
$\sigma_{\tilde{u}}^{\max}$	Maximal admissible value for $\hat{\sigma}_{\tilde{u}}^k$
$\sigma_{\tilde{y},k}^{(\tilde{u})}$	Approximate derivative of $\hat{\sigma}_{\tilde{y}}^k$ with respect to $\hat{\sigma}_{\tilde{u}}^k$
$\hat{\psi}_k$	Conjugate gradient update direction

Preliminary reading: Sections 2.2, 2.3, 2.4.3.

3.1 Introduction

In many applications, it is essential to identify a system online while the process which is generating the data is running. This requires recursive system identification algorithms (see. e.g. Ljung & Söderström 1983), which update an existing model as soon as new measured data becomes available. When not only the outputs, but also the inputs of the

system are corrupted by additive measurement noise, an errors-in-variables (EIV) setup (Söderström 2007*b*) arises. Within the literature only limited attention has been given to the development of recursive EIV identification algorithms. In particular, apart from the work of the author, no recursive approaches exist to identify an EIV system via the so-called Frisch scheme (Beghelli et al. 1990, Diversi et al. 2006, Söderström 2007*a*). The Frisch scheme is an EIV system identification technique, which yields not only the system parameters, but also the variances of the input and output measurement noise sequences. Consequently, this technique may be of interest for fault detection and condition monitoring purposes, since a change in the noise variances could, for instance, indicate faults associated with sensor devices. Hence, there is a natural motivation to develop a recursive identification algorithm based on the Frisch scheme.

This chapter develops recursive algorithms based on the offline Frisch scheme, which makes use of the Yule-Walker (YW) model selection criterion (cf. Section 2.4.3). The common idea is to use iterative procedures which carry out a single iteration as new data arrives. Such approaches are commonly utilised for recursive identification (see e.g. Ljung & Söderström 1983, Ljung 1999). The overall problem of developing recursive Frisch scheme algorithms can be divided into three subproblems:

1. Recursive computation of the parameter vector.
2. Recursive computation of the output measurement noise variance.
3. Recursive computation of the input measurement noise variance.

The first subproblem is addressed by taking the inconsistent recursive least squares (RLS) solution, from which the bias can be removed at each time instance, if an estimate of the input and output measurement noise variance is available. Such recursive bias compensating least squares (RBCLS) approaches are well-known within the literature (see e.g. Sagara & Wada 1977, Zheng & Feng 1989, Zheng 2000). The computation of the output measurement noise variance requires the solution of an eigenvalue problem. In order to obtain a recursive update equation, a conjugate gradient subspace tracking algorithm is utilised, which tracks the smallest eigenvalue of a slowly varying matrix. The third subproblem requires the minimisation of a model selection cost function. Two different approaches are discussed here: Firstly, a Gauss-Newton algorithm, which uses approximate derivatives is considered. Secondly, the nonlinear model selection criterion is replaced by an alternative cost function, which makes use of the linearised Frisch scheme equations. This allows a closed form solution of the input measurement noise variance to be obtained at each recursion step. The second approach is equivalent to the application of a steepest gradient technique in combination with a line search when use is made of the linearised Frisch scheme equations. Based on these two distinctively different methods for the determination of the input measurement noise variance, two different recursive Frisch scheme (RFS) algorithms are proposed, which are compared in simulation. Following on from this, a detailed analysis of the

computational costs of the derived algorithms is provided revealing the bottlenecks within the RFS computation. The absolute computation time per recursion of the two RFS algorithms is also compared to the offline case in a numerical example. Furthermore, attention is given to analyse the so-called Frisch-character, a feature which distinguishes the offline Frisch scheme from other identification techniques within the EIV framework. It is investigated via simulation, how the Frisch-character is affected when the recursive schemes are utilised to compute the estimates[1]. Finally, the RFS development is critically reviewed and alternative design strategies are discussed to overcome some of the potential shortcomings.

The chapter is organised as follows: Section 3.2 provides the formulation of the problem whilst the recursive algorithms are developed in Section 3.3. Their computational complexity is analysed in Section 3.4 and Section 3.5 is dedicated to investigate the Frisch-character of the resulting estimates. A critical discussion is provided in Section 3.6, and concluding remarks are given in Section 3.7.

Parts of this chapter have been published by the author in a series of papers: The core of the RFS development is described in (Linden, Vinsonneau & Burnham 2008) whilst early approaches, are reported in (Meyer, Linden, Vinsonneau & Burnham 2006, Linden, Meyer, Vinsonneau & Burnham 2006). Some analysis of the so-called Frisch-character of the estimates has been given in (Linden & Burnham 2008*c*), whilst the steepest gradient algorithm in combination with a line search has been proposed in (Linden, Larkowski & Burnham 2008).

3.2 Preliminaries

Typically, a recursive estimation scheme must obey the following principles (Ljung 1999):

P1 The processing must with certainty be completed during one sample interval using a fixed and a priori known amount of calculation.

P2 The data, which is passed from one recursion step to the next, must be stored in a finite-dimensional information vector.

The first principle is mainly associated with the computational complexity of the resulting algorithm. This aspect is fully investigated in Section 3.4. The second principle can be easily fulfilled for the Frisch-YW case, since it is straightforward to update the required covariance matrices $\hat{\Sigma}^k_{\bar{\varphi}}$ and $\hat{\Sigma}^k_{\zeta\bar{\varphi}}$ (see (2.47), (2.49)) or, in the case of adaptivity, the corresponding weighted arithmetic means (see Appendix B).

[1]Note that the Frisch-character is purely of academic interest and is investigated here to compare the recursive estimates with the offline estimates. It does, however, not reveal any information about the accuracy of the estimates obtained. With respect to practical applications, the Frisch-character might even be considered to be insignificant.

Recall from Section 2.4.3, that the estimates of the non-recursive Frisch-YW at time instance k are obtained by solving the nonlinear set of equations given by

$$\hat{\theta}_k = \left(\hat{\Sigma}_{\varphi}^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k)\right)^{-1} \hat{\xi}_{\varphi y}^k, \tag{3.1a}$$

$$\hat{\sigma}_{\tilde{y}}^k = \lambda_{\min}\left(\hat{A}_k\right), \tag{3.1b}$$

$$\hat{\sigma}_{\tilde{u}}^k = \arg\ \min_{\sigma_{\tilde{u}}} V_k, \tag{3.1c}$$

where

$$\Sigma_{\tilde{\varphi}}(\hat{\sigma}_k) = \begin{bmatrix} \hat{\sigma}_{\tilde{y}}^k I_{n_a} & 0 \\ 0 & \hat{\sigma}_{\tilde{u}}^k I_{n_b} \end{bmatrix}, \tag{3.2a}$$

$$\hat{A}_k \triangleq \hat{\Sigma}_{\bar{\varphi}_y}^k - \hat{\Sigma}_{\bar{\varphi}_y\varphi_u}^k \left[\hat{\Sigma}_{\varphi_u}^k - \hat{\sigma}_{\tilde{u}}^k I_{n_b}\right]^{-1} \hat{\Sigma}_{\varphi_u\bar{\varphi}_y}^k, \tag{3.2b}$$

$$V_k = \frac{1}{2}\|\hat{\Sigma}_{\zeta\bar{\varphi}}^k \bar{\theta}\|_2^2. \tag{3.2c}$$

Consequently, recursive expressions for (3.1) are to be developed which are outlined in the subsequent sections. Note that (3.1a) and (3.1b) form the core of the Frisch scheme and these are therefore termed the Frisch equations in the subsequent development. As in the offline Frisch scheme, the following assumptions are stated.

AS1 The dynamic system is asymptotically stable, i.e. $A(q^{-1})$ has all zeros inside the unit circle.

AS2 All system modes are observable and controllable, i.e. $A(q^{-1})$ and $B(q^{-1})$ have no common factors.

AS3 The polynomial degrees n_a and n_b are known a priori with $n_b \leq n_a$.

AI1 The true input u_{0_k} is a zero-mean ergodic process and is persistently exciting of sufficiently high order.

AN1 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are zero-mean, ergodic, white noises with unknown variances, denoted $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, respectively, i.e.

$$\sigma_{\tilde{u}}\delta_{kl} \triangleq E\left[\tilde{u}_k\tilde{u}_l\right], \tag{3.3a}$$

$$\sigma_{\tilde{y}}\delta_{kl} \triangleq E\left[\tilde{y}_k\tilde{y}_l\right]. \tag{3.3b}$$

AN2 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are mutually uncorrelated and also uncorrelated with both u_{0_k} and y_{0_k}.

3.3 Algorithmic development

This section develops recursive expressions for the Frisch scheme algorithm given by (3.1) based on the development presented in (Linden, Vinsonneau & Burnham 2008,

Linden, Larkowski & Burnham 2008). For the update of the estimated model parameters, a recursive bias-compensating least squares algorithm (RBCLS) is considered. Such approaches are well-known within the literature (see e.g. Sagara & Wada 1977) and have also been applied to iteratively solve the (offline) bias compensation problem (Zheng & Feng 1989, Zheng 2000). The underlying idea is to compute the RLS solution and to compensate for the asymptotic bias at each time instance. In order to compensate for the bias, the variances of the input and output measurement noise sequences are required, for which gradient-based approaches are considered here. Note that the recursive computation of the variances, which is based on the Frisch scheme approach, distinguishes the algorithms developed within this chapter from other recursive/iterative bias compensating schemes within the literature. Recall from (3.1b), that an estimate of the output measurement noise variance is determined by solving an eigenvalue problem. This can be (approximately) solved recursively, by making use of a conjugate gradient method, which tracks the smallest eigenvalue of a slowly varying matrix. For the update of the input measurement noise estimate two possibilities are considered within this section: Whilst the initial development in (Linden, Vinsonneau & Burnham 2008) considered a steepest gradient algorithm with fixed step size, here, the minimum of the YW model selection cost function (3.2c) is updated either via a Gauss-Newton algorithm, or a steepest gradient search employing a line search facility. The latter is equivalent to minimising a modified cost function, which exclusively depends on linearisations of the Frisch scheme equations (3.1a)-(3.1b) and which allows a closed form solution of $\hat{\sigma}_{\tilde{u}}^k$ to be obtained at each time instance k. This approach has been proposed in (Linden, Larkowski & Burnham 2008).

Firstly, the update of the covariance matrices is discussed in Section 3.3.1 followed by development of the update equation for θ, $\sigma_{\tilde{y}}$ and $\sigma_{\tilde{u}}$ in Sections 3.3.2-3.3.4, respectively. The overall algorithms are summarised in Section 3.3.5 followed by a numerical example, which is given in Section 3.3.6.

3.3.1 Update of covariance matrices

In order to satisfy requirement **P2**, the covariance elements $\hat{\Sigma}_{\varphi}^k$, $\hat{\xi}_{\varphi y}^k$, $\hat{\Sigma}_{\bar{\varphi}_y}^k$, $\hat{\Sigma}_{\bar{\varphi}_y\varphi_u}^k$, $\hat{\Sigma}_{\varphi_u}^k$, $\hat{\Sigma}_{\varphi_u\bar{\varphi}_y}^k$ and $\hat{\Sigma}_{\zeta\bar{\varphi}}^k$ in (3.1) are to be updated. Since the first six covariance elements are contained within (cf. the definitions (2.10)-(2.12))

$$\hat{\Sigma}_{\bar{\varphi}}^k = \begin{bmatrix} \hat{\Sigma}_{\bar{\varphi}_y}^k & \hat{\Sigma}_{\bar{\varphi}_y\varphi_u}^k \\ \hat{\Sigma}_{\varphi_u\bar{\varphi}_y}^k & \hat{\Sigma}_{\varphi_u}^k \end{bmatrix} = \begin{bmatrix} \hat{r}_y(0) & -\hat{\xi}_{y\varphi}^k \\ -\hat{\xi}_{\varphi y}^k & \hat{\Sigma}_{\varphi}^k \end{bmatrix}, \tag{3.4}$$

it is sufficient to consider the update of $\hat{\Sigma}_{\bar{\varphi}}^k$ and $\hat{\Sigma}_{\zeta\bar{\varphi}}^k$. The covariance matrices can be updated in a straightforward manner as

$$\hat{\Sigma}_{\bar{\varphi}}^k = \hat{\Sigma}_{\bar{\varphi}}^{k-1} + \frac{1}{k}\left(\bar{\varphi}_k\bar{\varphi}_k^T - \hat{\Sigma}_{\bar{\varphi}}^{k-1}\right), \tag{3.5a}$$

$$\hat{\Sigma}_{\zeta\bar{\varphi}}^k = \hat{\Sigma}_{\zeta\bar{\varphi}}^{k-1} + \frac{1}{k}\left(\zeta_k\bar{\varphi}_k^T - \hat{\Sigma}_{\zeta\bar{\varphi}}^{k-1}\right). \tag{3.5b}$$

Using (3.5), it is now possible to evaluate the Frisch-YW equations (3.1) at each time step. Although such an algorithm may satisfy **P1** and **P2** (provided the time used for the minimisation of V_k is limited), it cannot be considered to be a recursive scheme since only the trivial covariance matrix update operations are performed. However, such a repeatedly applied Frisch scheme (RAFS) is used for comparison purposes in the subsequent development, since it exhibits all the characteristic properties of the Frisch scheme. The algorithm can be summarised as follows.

Algorithm 3.1 (RAFS).

1) Initialisation

2) For $k = n_\zeta + n_b + 1, ...$

- **a)** Update $\hat{\Sigma}_{\bar{\varphi}}^k$ and $\hat{\Sigma}_{\zeta\bar{\varphi}}^k$ via (3.5)
- **b)** Compute $\hat{\sigma}_{\tilde{u}}^k$ via (3.1c)
- **c)** Determine $\hat{\sigma}_{\tilde{y}}^k$ by solving (3.1b)
- **d)** Compute $\hat{\theta}_k$ via (3.1a)

In order to compute $\hat{\sigma}_{\tilde{u}}^k$, an optimisation has to be performed to obtain the minimum of the YW cost function, which is usually achieved in an iterative way. If Matlab is used to implement the algorithm, readily available routines such as `fminsearch`, which applies the Nelder-Mead simplex method (cf. MathWorks 2007), can be utilised. Recall that in order to evaluate the value of the cost function, (3.1a) and (3.1b) need to be computed, which means that an eigenvalue problem has to be solved at each iteration step (several times at each k). The computational burden of such an optimisation, which is to be carried out at each recursion step of the RAFS, is therefore expected to be rather high. In order to overcome this shortcoming, and following the philosophy of recursive identification techniques, it is possible to apply only one iteration per recursion, i.e to set the number of maximal iterations to unity. This will, if the optimisation is initialised with the most recent value of the input measurement noise variance estimate, gradually improve the estimate of $\sigma_{\tilde{u}}$ as time evolves.

Adaptivity

In order to equip the algorithm with some form of adaptivity, exponential data weighting (forgetting) may be utilised within the update of the covariance matrices. Therefore, one common approach is to assume that the process generating the data is varying slowly over time such that stationarity can be assumed to hold approximately (DeGroat, Dowling & Linebarger 1997). Other choices of data weighting, such as moving windows, might be more appropriate depending on the desired tracking capability of the estimator. This is, however, not considered here.

In the case of exponential data weighting, the covariance matrices become weighted arithmetic means, which are denoted $\bar{\Sigma}_{\bar{\varphi}}^k$ and $\bar{\Sigma}_{\zeta\bar{\varphi}}^k$. By weighting the ith data at time k with

$$\beta_i^k = \lambda_k \beta_i^{k-1} \quad \text{for} \quad 0 \leq i \leq k-1 \quad \text{and} \quad \beta_k^k \triangleq 1, \tag{3.6}$$

where λ_k denotes the forgetting factor, the general update equations for the covariance matrices are given by (cf. Chapter 11.2 in (Ljung 1999) or see Appendix B for details)

$$\bar{\Sigma}_{\bar{\varphi}}^k = \bar{\Sigma}_{\bar{\varphi}}^{k-1} + \gamma_k \left(\bar{\varphi}_k \bar{\varphi}_k^T - \bar{\Sigma}_{\bar{\varphi}}^{k-1} \right), \tag{3.7a}$$

$$\bar{\Sigma}_{\zeta\bar{\varphi}}^k = \bar{\Sigma}_{\zeta\bar{\varphi}}^{k-1} + \gamma_k \left(\zeta_k \bar{\varphi}_k^T - \bar{\Sigma}_{\zeta\bar{\varphi}}^{k-1} \right). \tag{3.7b}$$

The normalising gain γ_k is given by

$$\gamma_k \triangleq \left(\sum_{i=1}^{k} \beta_{k,i} \right)^{-1} = \frac{\gamma_{k-1}}{\lambda_k + \gamma_{k-1}}, \tag{3.8}$$

which reduces to $1/k$ in the case of no adaptivity, i.e. λ_k equal to 1. In the case of exponential forgetting, i.e. $\lambda_k = \lambda$ where $0 < \lambda < 1$, the normalising gain becomes $1 - \lambda$.

Rather than using an exponential weighting of the data, it is possible to describe the variation of the parameters in a stochastic manner which is frequently done by modelling the variation as random walks or generalisations. If the parameters are assumed to vary in a more rapid manner, the system might be described via state dependent parameter models. The identification of such systems is the topic of (Young, McKenna & Bruun 2001). This is, however, not further considered here, and for simplicity, the case of no forgetting ($\gamma_k = 1/k$) is considered in the subsequent development (which implies that the $\hat{}$ notation $\hat{\Sigma}_{\bar{\varphi}}^k$ and $\hat{\Sigma}_{\zeta\bar{\varphi}}^k$ is used).

The next subsection provides recursive update equations of the system parameter estimates.

3.3.2 Update of θ

If the output measurement noise $\sigma_{\tilde{y}}$ was computed via (3.1b), i.e. the eigenvalue problem was solved exactly, the resulting set of normal equations

$$\left(\hat{\Sigma}_{\varphi}^{k} - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k)\right)\hat{\theta}_k = \hat{\xi}_{\varphi y} \tag{3.9}$$

would be consistent and could be uniquely solved for $\hat{\theta}_k$, e.g. via Gaussian elimination. In the case that the eigenvalue problem is solved approximately (as discussed in Section 3.3.3) the normal equations are inconsistent and an approximate solution is required to be searched for. Whilst other choices are possible (e.g. total least squares), the most straightforward way to compute $\hat{\theta}_k$ would be via the usage of least squares (LS). In order to obtain a recursive expression for $\hat{\theta}_k$, an approach is adopted here, similar to that in (Sagara & Wada 1977, Zheng & Feng 1989), where the bias of the recursive least squares (RLS) estimate is compensated at each time step k. For now, it is assumed that estimates of $\hat{\sigma}_{\tilde{u}}^{k}$ and $\hat{\sigma}_{\tilde{y}}^{k}$ have already been obtained. The update equations for the latter two quantities are developed in the remainder of this Section (see Sections 3.3.3 and 3.3.4 below).

Using the RBCLS approach of Section 2.4.1, allows a recursive estimate of θ to be obtained as summarised in the following algorithm (cf. also Appendix A for a more detailed derivation).

Algorithm 3.2 (RBCLS).

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{LS}} + P_k \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k)\hat{\theta}_{k-1}. \tag{3.10}$$

The corresponding RLS equations for the determination of $\hat{\theta}_k^{\mathrm{LS}}$ and P_k are given as follows.

Algorithm 3.3 (Normalised gain RLS).

$$\hat{\theta}_k^{\mathrm{LS}} = \hat{\theta}_{k-1}^{\mathrm{LS}} + L_k\left(y_k - \varphi_k^T \hat{\theta}_{k-1}^{\mathrm{LS}}\right) \tag{3.11a}$$

$$L_k = \frac{P_{k-1}\varphi_k}{\varphi_k^T P_{k-1}\varphi_k + \frac{1-\gamma_k}{\gamma_k}} \tag{3.11b}$$

$$P_k = \frac{1}{1-\gamma_k}\left(P_{k-1} - \frac{P_{k-1}\varphi_k\varphi_k^T P_{k-1}}{\varphi_k^T P_{k-1}\varphi_k + \frac{1-\gamma_k}{\gamma_k}}\right) \tag{3.11c}$$

Note that

$$P_k = [\hat{\Sigma}_\varphi^k]^{-1}, \tag{3.12}$$

which means that by making use of the above normalised gain version of RLS and applying the matrix inversion lemma (see Ch. 11.2 in Ljung 1999), utilisation of P_k in (3.10) to compute $[\hat{\Sigma}_\varphi^k]^{-1}$ recursively, conveniently avoids the need for matrix inversion.

Remark 3.1. The RLS algorithm provides another means to introduce adaptivity into the identification via exponential data weighting controlled by γ_k in (3.11). By tuning the gain γ_k in (3.7) (which is generally different to that used in (3.11)), the user can choose whether to introduce adaptivity for the estimation of the variances as well as the parameters, or solely for the system parameters (the latter can be achieved by setting γ_k in (3.7) to $1/k$ whilst choosing $\gamma_k = 1 - \lambda_k$ in (3.11)).

The next subsection considers the recursive computation of the output measurement noise variance.

3.3.3 Update of $\sigma_{\tilde{y}}$

In order to compute $\hat{\sigma}_{\tilde{y}}^k$, the least eigenvalue of $\hat{A}_k$ in (3.2b) is required to be determined, which generally requires $O(n^3)$ flops for a $n \times n$ matrix (Golub & Van Loan 1996). When only a few eigenpairs are required, more efficient algorithms exist which only track the subspace corresponding to one or more eigenvalues (Comon & Golub 1990). Such an approach is feasible when the corresponding matrix (hence the singular triplets) 'varies slowly' with time, which is assumed to be the case here, provided the estimate $\hat{\sigma}_{\tilde{u}}^k$ does not exhibit any rapid changes[2]. Therefore, introduce the following assumption[3].

AE1 The estimate of the input measurement noise variance $\hat{\sigma}_{\tilde{u}}^k$ 'varies slowly' with time.

Since the matrix update $\Delta A_k \triangleq \hat{A}_k - \hat{A}_{k-1}$ is generally of full rank $n_a + 1$, a gradient based algorithm, which requires $O(n^2)$ flops[4], is applied in the subsequent development, in order to determine a recursive expression for $\hat{\sigma}_{\tilde{y}}^k$. More specifically, an iterative conjugate gradient method similar to that proposed in (Chen, Sarkar, Dianat & Brulé 1986, Feng & Owen 1996, Yang 1993) is used, where one iteration per recursion is applied.

Remark 3.2 (Choice of subspace tracking algorithm). Note that tracking eigenpairs or singular triplets is a common problem in the area of signal processing and the corresponding research area is termed subspace tracking, which has experienced tremendous

[2]Note that the sample covariance elements contained in $\hat{A}_k$ converge towards their expected values for an increasing number of samples due to the stated assumptions.

[3]Here, **AE** corresponds to the assumptions concerning the estimator.

[4]In the case of a rank-one update, it is possible to track d singular triplets using $O(nd^2)$ flops only (cf. Section V in (Comon & Golub 1990) or see (Davila 1994)).

interest in the literature (see (Comon & Golub 1990) for a detailed survey and (DeGroat et al. 1997) for the developments since 1990). Consequently, there exists a rich collection of algorithms to tackle the problem of tracking $\hat{\sigma}_{\tilde{y}}$ within the recursive Frisch-YW algorithm. Whilst in this chapter the approach is focused on the conjugate gradient methods, other choices (e.g. inverse power iteration) might well be possible.

Suppose the eigensystem is given by

$$\hat{A}_k x_k = \sigma_{\tilde{y}}^k x_k, \tag{3.13}$$

where $x_k \in \mathbb{R}^{n_a+1}$ denotes an eigenvector of $\hat{A}_k$ and $\sigma_{\tilde{y}}^k \in \mathbb{R}$ the corresponding eigenvalue. Then, the minimum eigenvalue can be obtained by minimising the Rayleigh quotient (Golub & Van Loan 1996)

$$\sigma_{\tilde{y}}^k = R(x_k) \triangleq \frac{x_k^T \hat{A}_k x_k}{x_k^T x_k}. \tag{3.14}$$

Utilising a conjugate gradient method to minimise $R(x_k)$, the update equations for the minimum eigenvalue are given by (cf. Feng & Owen 1996)

$$\hat{x}_k = \hat{\bar{x}}_{k-1} + \hat{\mu}_{k-1}\hat{\psi}_{k-1}, \tag{3.15a}$$
$$\hat{\bar{x}}_k = \hat{x}_k / \left(\hat{x}_k^T \hat{x}_k\right)^{1/2}, \tag{3.15b}$$
$$\hat{\sigma}_{\tilde{y}}^k = \hat{\bar{x}}_k^T \hat{A}_k \hat{\bar{x}}_k, \tag{3.15c}$$

where $\hat{x}_k$ denotes an estimate of the eigenvector x_k and $\hat{\bar{x}}_k$ is a scaled version of unity length. The scalar $\hat{\mu}_k$ denotes the stepsize whilst $\hat{\psi}_k \in \mathbb{R}^{n_a+1}$ denotes the conjugate gradient update direction which is given by

$$\hat{r}_k = \hat{A}_k \hat{\bar{x}}_k - \hat{\sigma}_{\tilde{y}}^k \hat{\bar{x}}_k, \tag{3.16a}$$
$$\hat{q}_{k-1} = -(\hat{r}_k^T \hat{A}_k \hat{\psi}_{k-1})/(\hat{\psi}_{k-1}^T \hat{A}_k \hat{\psi}_{k-1}), \tag{3.16b}$$
$$\hat{\psi}_k = \hat{r}_k + \hat{q}_{k-1}\hat{\psi}_{k-1}, \tag{3.16c}$$

where $\hat{r}_k$ denotes the residual. The optimal step size is chosen as

$$\hat{\mu}_k = \left(\hat{\sigma}_{\tilde{y}}^k d_4^k - d_2^k + \sqrt{d_5^k}\right) / \left(2\left(d_2^k d_3^k - d_1^k d_4^k\right)\right), \tag{3.17}$$

where

$$d_1^k = \bar{x}_k^T \hat{A}_k \hat{\psi}_k, \qquad d_2^k = \hat{\psi}_k^T \hat{A}_k \hat{\psi}_k,$$
$$d_3^k = \bar{x}_k^T \hat{\psi}_k, \qquad d_4^k = \hat{\psi}_k^T \hat{\psi}_k \tag{3.18}$$

and

$$d_5^k = \left(\hat{\sigma}_{\tilde{y}}^k d_4^k - d_2^k\right)^2 - 4\left(d_2^k d_3^k - d_1^k d_4^k\right)\left(d_1^k - \hat{\sigma}_{\tilde{y}}^k d_3^k\right). \tag{3.19}$$

With optimal step size it is meant that a line search (cf. Dennis & Schnabel 1996, Section 6.3) is performed, which chooses the step size $\hat{\mu}_k$ in a manner, such that the Rayleigh quotient is minimised within the given search direction $\hat{\psi}_k$. The algorithm is initialised with a guess of $\hat{x}_0$ and $\hat{A}_0$, such that

$$\hat{\bar{x}}_0 = \hat{x}_0 / \left(\hat{x}_0^T \hat{x}_0\right)^{1/2}, \tag{3.20a}$$

$$\hat{\psi}_0 = \hat{r}_0 = \hat{A}_0 \hat{\bar{x}}_0 - R(\hat{x}_0)\hat{\bar{x}}_0. \tag{3.20b}$$

Remark 3.3 (Frisch-like character). One of the significant characteristics of the Frisch scheme is that the estimated model belongs to a class characterised by a convex curve in the noise space (see Figure 2.2), where the functional relationship between $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$ defining a locus of solutions is given by (3.1b). This feature has been termed the Frisch-character in Definition 2.1. Computing the output noise variance recursively via (3.15)-(3.20) will inevitably introduce an error, which means that the estimated set $(\hat{\sigma}_{\tilde{u}}^k, \hat{\sigma}_{\tilde{y}}^k)$ will not exactly lie on this convex curve, i.e. the Frisch-character will only be approximated. This would, strictly speaking, only imply a Frisch-like character for the solution. However, as will be illustrated in Section 3.3.6, the set $(\hat{\sigma}_{\tilde{u}}^k, \hat{\sigma}_{\tilde{y}}^k)$ can converge to the convex curve, after the initialisation transients have decayed.

The conjugate gradient subspace tracking algorithm, which minimises the Rayleigh quotient, is denoted CG-RQ and summarised as follows.

Algorithm 3.4 (CG-RQ).

$$\hat{\sigma}_{\tilde{y}}^k = \hat{\bar{x}}_k^T \hat{A}_k \hat{\bar{x}}_k \tag{3.21a}$$

$$\hat{\bar{x}}_k = \hat{x}_k / \left(\hat{x}_k^T \hat{x}_k\right)^{1/2} \tag{3.21b}$$

$$\hat{A}_k = \hat{\Sigma}_{\bar{\varphi}_y}^k - \hat{\Sigma}_{\bar{\varphi}_y \varphi_u}^k \left[\hat{\Sigma}_{\varphi_u}^k - \hat{\sigma}_{\tilde{u}}^k I_{n_b}\right]^{-1} \hat{\Sigma}_{\varphi_u \bar{\varphi}_y}^k \tag{3.21c}$$

$$\hat{x}_k = \hat{\bar{x}}_{k-1} + \hat{\mu}_{k-1}\hat{\psi}_{k-1} \tag{3.21d}$$

$$\hat{\mu}_k = \left(\hat{\sigma}_{\tilde{y}}^k d_4^k - d_2^k + \sqrt{d_5^k}\right) / \left(2\left(d_2^k d_3^k - d_1^k d_4^k\right)\right) \tag{3.21e}$$

$$d_1^k = \hat{\bar{x}}_k^T \hat{A}_k \hat{\psi}_k \tag{3.21f}$$

$$d_2^k = \hat{\psi}_k^T \hat{A}_k \hat{\psi}_k \tag{3.21g}$$

$$d_3^k = \hat{\bar{x}}_k^T \hat{\psi}_k \tag{3.21h}$$

$$d_4^k = \hat{\psi}_k^T \hat{\psi}_k \tag{3.21i}$$

$$d_5^k = \left(\hat{\sigma}_{\tilde{y}}^k d_4^k - d_2^k\right)^2 - 4\left(d_2^k d_3^k - d_1^k d_4^k\right)\left(d_1^k - \hat{\sigma}_{\tilde{y}}^k d_3^k\right) \tag{3.21j}$$

$$\hat{\psi}_k = \hat{r}_k + \hat{q}_{k-1}\hat{\psi}_{k-1} \tag{3.21k}$$
$$\hat{r}_k = \hat{A}_k\hat{\bar{x}}_k - \hat{\sigma}^k_{\tilde{y}}\hat{\bar{x}}_k \tag{3.21l}$$
$$\hat{q}_{k-1} = -(\hat{r}_k^T\hat{A}_k\hat{\psi}_{k-1})/(\hat{\psi}_{k-1}^T\hat{A}_k\hat{\psi}_{k-1}) \tag{3.21m}$$

Consider now the recursive computation of the input measurement noise variance.

3.3.4 Update of $\sigma_{\tilde{u}}$

For the recursive computation of $\hat{\sigma}^k_{\tilde{u}}$, two algorithms are developed within this subsection. The first algorithm considers a Gauss-Newton approach, whilst the second algorithm minimises a modified cost function, based on the linearised Frisch equations, for which a closed-form solution can be obtained at each time step.

Yule-Walker cost function

Recall that within the Frisch scheme, an estimate of the input measurement noise variance $\sigma_{\tilde{u}}$ can be obtained by minimising the Yule-Walker (YW) model selection cost function (3.2c), which can be expressed as a nonlinear least squares (NLS) problem defined by

$$\hat{\sigma}^k_{\tilde{u}} = \arg\min_{\sigma_{\tilde{u}}} V_k, \tag{3.22a}$$
$$V_k = \frac{1}{2}\left\|r_k(\theta)\right\|_2^2, \tag{3.22b}$$

where $r_k(\theta)$ denotes the NLS residual which is defined by

$$r_k(\theta) \triangleq \begin{bmatrix} r_1^k \\ \vdots \\ r_{n_\theta}^k \end{bmatrix} = \begin{bmatrix} -\hat{\xi}_{\zeta y} & \hat{\Sigma}^k_{\zeta\varphi} \end{bmatrix}\begin{bmatrix} 1 \\ \theta \end{bmatrix} = \hat{\Sigma}^k_{\zeta\varphi}\theta - \hat{\xi}^k_{\zeta y}. \tag{3.23}$$

Equation (3.22b) can be minimised by making use of a general Newton method given by (cf. Ljung 1999, p. 326)

$$\hat{\sigma}^k_{\tilde{u}} = \hat{\sigma}^{k-1}_{\tilde{u}} - \gamma_k\left[V_k''\right]^{-1}V_k', \tag{3.24}$$

where V_k' and V_k'' denote the first and second order derivative of the YW cost function (3.22b), respectively, whilst γ_k is a scalar step size. If the second order derivative is approximated, a Gauss-Newton method is obtained. In order to proceed, it is necessary

to obtain the derivative

$$V_k' \triangleq \frac{d}{d\hat{\sigma}_{\tilde{u}}^k} V_k \tag{3.25}$$

at each recursion step. Recall that for the evaluation of the cost function V_k, the θ-equation (3.1a) and the $\lambda_{\min}$-equation (3.1b) are required as well. Hence, V_k is a nonlinear function of $\sigma_{\tilde{u}}$ and its exact evaluation at each time instance k is not desired (since this would imply the need to solve an eigenvalue problem at each time step). For this reason, an approximate gradient, based on linearised versions of (3.1a) and (3.1b), is considered in the subsequent development.

Linearisation of the Frisch equations

Linearised expressions of the Frisch equations (3.1a) and (3.1b) have been derived in (Söderström 2007*a*) and are given by the following lemma.

Lemma 3.1 (Linearised Frisch equations). Carrying out the linearisation of (3.1a) and (3.1b) around the point

$$\vartheta_* \triangleq \begin{bmatrix} \theta_*^T & \sigma_{\tilde{y}}^* & \sigma_{\tilde{u}}^* \end{bmatrix}^T = \begin{bmatrix} a_*^T & b_*^T & \sigma_{\tilde{y}}^* & \sigma_{\tilde{u}}^* \end{bmatrix}^T, \tag{3.26}$$

the linearised Frisch equations are given by

$$\hat{\theta}_k \approx L_\theta(\vartheta_*) \triangleq \theta_* + \left(\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\sigma_*)\right)^{-1} \left(\hat{\xi}_{\varphi y}^k - \hat{\Sigma}_\varphi^k \theta_* + \begin{bmatrix} \hat{\sigma}_{\tilde{y}}^k a_* \\ \hat{\sigma}_{\tilde{u}}^k b_* \end{bmatrix}\right), \tag{3.27a}$$

$$\hat{\sigma}_{\tilde{y}}^k \approx L_{\sigma_{\tilde{y}}}(\vartheta_*) \triangleq \sigma_{\tilde{y}}^* + \frac{b_*^T b_*}{\bar{a}_*^T \bar{a}_*} \left(\sigma_{\tilde{u}}^* - \hat{\sigma}_{\tilde{u}}^k\right), \tag{3.27b}$$

where it is assumed that ϑ_* is close to ϑ.

Proof. See Appendix C for a detailed derivation. ∎

Also introduce

$$\iota(\vartheta_*) \triangleq \hat{\xi}_{\varphi y}^k - \hat{\Sigma}_\varphi^k \theta_* + \begin{bmatrix} \sigma_{\tilde{y}}^* + \frac{b_*^T b_*}{\bar{a}_*^T \bar{a}_*} \sigma_{\tilde{u}}^* \end{bmatrix} \begin{bmatrix} a_* \\ 0 \end{bmatrix}, \tag{3.28a}$$

$$\kappa(\vartheta_*) \triangleq \begin{bmatrix} -\frac{b_*^T b_*}{\bar{a}_*^T \bar{a}_*} a_* \\ b_* \end{bmatrix}, \tag{3.28b}$$

$$\Sigma_{\varphi_0}(\sigma_*) \triangleq \hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\sigma_*). \tag{3.28c}$$

Using this notation, the quantity $\sigma_{\tilde{y}}$ given by (3.27b) can be eliminated in (3.27a) yielding a linear expression for θ which only depends on $\hat{\sigma}_{\tilde{u}}^k$

$$L_\theta(\vartheta_*) = \theta_* + \Sigma_{\varphi_0}^{-1}(\sigma_*)\iota(\vartheta_*) + \Sigma_{\varphi_0}^{-1}(\sigma_*)\kappa(\vartheta_*)\hat{\sigma}_{\tilde{u}}^k. \tag{3.29}$$

Equation (3.29) will be utilised later, for the development of the second algorithm.

Computation of the derivative

By applying the chain rule for vector differentiation, the exact derivative of V_k is given by

$$V_k' \triangleq \frac{dV_k}{d\hat{\sigma}_{\tilde{u}}^k} = \frac{dV_k}{d\theta}\left(\frac{\partial\theta}{\partial\sigma_{\tilde{y}}}\frac{d\sigma_{\tilde{y}}}{d\sigma_{\tilde{u}}} + \frac{\partial\theta}{\partial\sigma_{\tilde{u}}}\right)\Bigg|_{\vartheta=\hat{\vartheta}_k} \in \mathbb{R}, \tag{3.30}$$

where $\frac{dV_k}{d\theta} \in \mathbb{R}^{1\times n_\theta}$, $\frac{\partial\theta}{\partial\sigma_{\tilde{y}}} \in \mathbb{R}^{n_\theta}$, $\frac{\partial\theta}{\partial\sigma_{\tilde{u}}} \in \mathbb{R}^{n_\theta}$ and $\frac{d\sigma_{\tilde{y}}}{d\sigma_{\tilde{u}}} \in \mathbb{R}$. The task is now to determine approximate expressions for the individual derivative terms in (3.30). Using (3.23) the cost function (3.22b) is re-expressed as

$$\begin{aligned} V_k &= \frac{1}{2}\left(\hat{\Sigma}_{\zeta\varphi}^k\theta - \hat{\xi}_{\zeta y}\right)^T\left(\hat{\Sigma}_{\zeta\varphi}^k\theta - \hat{\xi}_{\zeta y}\right) \\ &= \frac{1}{2}\left(\hat{\xi}_{\zeta y}^{k^T}\hat{\xi}_{\zeta y}^k - \hat{\xi}_{\zeta y}^{k^T}\hat{\Sigma}_{\zeta\varphi}^k\theta - \theta^T\hat{\Sigma}_{\zeta\varphi}^{k^T}\hat{\xi}_{\zeta y}^k + \theta^T\hat{\Sigma}_{\zeta\varphi}^{k^T}\hat{\Sigma}_{\zeta\varphi}^k\theta\right), \end{aligned} \tag{3.31}$$

and using the rules for vector differentiation, one obtains for the first term in (3.30)

$$\frac{dV_k}{d\theta} = \theta^T\hat{\Sigma}_{\zeta\varphi}^{k^T}\hat{\Sigma}_{\zeta\varphi}^k - \hat{\xi}_{\zeta y}^{k^T}\hat{\Sigma}_{\zeta\varphi}^k \in \mathbb{R}^{1\times n_\theta}. \tag{3.32}$$

At time instance k, the parameter vector θ may be approximated with $\hat{\theta}_{k-1}$, which gives the approximate derivative

$$\frac{dV_k}{d\theta} \approx V_k^{(\theta)} \triangleq \left(\hat{\theta}_{k-1}^T\hat{\Sigma}_{\zeta\varphi}^{k^T} - \hat{\xi}_{\zeta y}^{k^T}\right)\hat{\Sigma}_{\zeta\varphi}^k. \tag{3.33}$$

In addition, expressions for the sensitivity derivatives $\partial\theta/\partial\sigma_{\tilde{u}}$, $\partial\theta/\partial\sigma_{\tilde{y}}$ and $d\sigma_{\tilde{y}}/d\sigma_{\tilde{u}}$, evaluated at $\vartheta = \hat{\vartheta}_k$, are given by the following lemma.

Lemma 3.2. The approximate sensitivity derivatives are given by

$$\frac{\partial\hat{\theta}_k}{\partial\hat{\sigma}_{\tilde{y}}^k} \approx \theta_k^{(\tilde{y})} = \left(\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right)^{-1}\begin{bmatrix}\hat{a}_{k-1}\\ 0\end{bmatrix} \in \mathbb{R}^{n_\theta}, \tag{3.34a}$$

$$\frac{\partial\hat{\theta}}{\partial\hat{\sigma}_{\tilde{u}}^k} \approx \theta_k^{(\tilde{u})} = \left(\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right)^{-1}\begin{bmatrix}0\\ \hat{b}_{k-1}\end{bmatrix} \in \mathbb{R}^{n_\theta}, \tag{3.34b}$$

$$\frac{d\hat{\sigma}_{\tilde{y}}^k}{d\hat{\sigma}_{\tilde{u}}^k} \approx \sigma_{\tilde{y},k}^{(\tilde{u})} = -\frac{\hat{b}_{k-1}^T\hat{b}_{k-1}}{\hat{\hat{a}}_{k-1}^T\hat{\hat{a}}_{k-1}} \in \mathbb{R}. \tag{3.34c}$$

Proof. Assuming that $\hat{\vartheta}_{k-1}$ is close to $\hat{\vartheta}_k$, the sensitivity derivatives can be directly computed from the linearised Frisch equations (3.27) when the linearisation is carried out around $\vartheta_* = \hat{\vartheta}_{k-1}$. ∎

Consequently, the gradient of the YW cost function with respect to $\hat{\sigma}_{\tilde{u}}^k$ can be computed by (3.30) using the approximations (3.33) and (3.34), which defines the (approximate) gradient update direction as

$$\begin{aligned} V_k' \approx V_k^{(\sigma_{\tilde{u}})} &= V_k^{(\theta)} \left(\theta_k^{(\tilde{y})} \sigma_{\tilde{y},k}^{(\tilde{u})} + \theta_k^{(\tilde{u})} \right) \\ &= \left(\hat{\theta}_{k-1}^T \hat{\Sigma}_{\zeta\varphi}^{k\,T} - \hat{\xi}_{\zeta y}^{k\,T} \right) \hat{\Sigma}_{\zeta\varphi}^k \left(\hat{\Sigma}_{\varphi}^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \right)^{-1} \begin{bmatrix} -\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} \hat{a}_{k-1} \\ \hat{b}_{k-1} \end{bmatrix}, \end{aligned} \tag{3.35}$$

or equivalently

$$V_k^{(\sigma_{\tilde{u}})} = \left[-\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} \hat{a}_{k-1}^T, \quad \hat{b}_{k-1}^T \right] \left(\hat{\Sigma}_{\varphi}^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \right)^{-1} \hat{\Sigma}_{\zeta\varphi}^{k\,T} \left(\hat{\Sigma}_{\zeta\varphi}^k \hat{\theta}_{k-1} - \hat{\xi}_{\zeta y}^k \right). \tag{3.36}$$

An alternative way to arrive at (3.36) would be to consider (3.22b) as a NLS problem, for which the gradient of V_k is given by

$$V_k' = J^T(\sigma_{\tilde{u}}) r_k(\theta), \tag{3.37}$$

where $J(\sigma_{\tilde{u}})$ denotes the Jacobian[5] defined by (cf. Björck 1996, p. 340)

$$J(\sigma_{\tilde{u}}) \triangleq \frac{\partial r_k(\theta)}{\partial \sigma_{\tilde{u}}} = \begin{bmatrix} \frac{\partial r_1^k}{\partial \sigma_{\tilde{u}}} \\ \vdots \\ \frac{\partial r_{n_\theta}^k}{\partial \sigma_{\tilde{u}}} \end{bmatrix}, \tag{3.38}$$

In view of (3.37), equation (3.36) can be expressed as

$$V_k^{(\sigma_{\tilde{u}})} = J_k^{(\sigma_{\tilde{u}})^T} r_k(\hat{\theta}_{k-1}), \tag{3.39}$$

where the approximate Jacobian $J_k^{(\sigma_{\tilde{u}})}$ and the approximate residual $r_k(\hat{\theta}_{k-1})$ are respectively given by

$$J_k^{(\sigma_{\tilde{u}})^T} = \left[-\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} \hat{a}_{k-1}^T, \quad \hat{b}_{k-1}^T \right] \left(\hat{\Sigma}_{\varphi}^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \right)^{-1} \hat{\Sigma}_{\zeta\varphi}^{k\,T}, \tag{3.40a}$$

$$r_k(\hat{\theta}_{k-1}) = \hat{\Sigma}_{\zeta\varphi}^k \hat{\theta}_{k-1} - \hat{\xi}_{\zeta y}^k. \tag{3.40b}$$

Note that two approximations have been utilised to arrive at the gradient:

1. The residual is approximated by making use of the most recent estimate of θ.
2. The Jacobian is obtained by making use of the linearised Frisch equations.

[5] Which reduces to a $1 \times n_\theta$ vector in this univariate case.

Gauss-Newton algorithm

After the approximate derivative of the YW cost function has been obtained, it is possible to design a Gauss-Newton algorithm to obtain a recursive update equation for $\hat{\sigma}_{\tilde{u}}^k$. Therefore, the second order derivative can approximated as (see Björck 1996, p. 340)

$$V_k'' \approx {J_k^{(\sigma_{\tilde{u}})}}^T J_k^{(\sigma_{\tilde{u}})}. \tag{3.41}$$

The (approximate) Gauss-Newton algorithm (see (3.24)), which minimises the YW cost function, is denoted YW-GN and can be summarised as follows.

Algorithm 3.5 (YW-GN).

$$\hat{\sigma}_{\tilde{u}}^k = \hat{\sigma}_{\tilde{u}}^{k-1} - \gamma_k \left[{J_k^{(\sigma_{\tilde{u}})}}^T J_k^{(\sigma_{\tilde{u}})} \right]^{-1} {J_k^{(\sigma_{\tilde{u}})}}^T r_k(\hat{\theta}_{k-1}) \tag{3.42a}$$

$${J_k^{(\sigma_{\tilde{u}})}}^T = \left[-\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} \hat{a}_{k-1}^T, \quad \hat{b}_{k-1}^T \right] \left(\hat{\Sigma}_{\varphi}^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \right)^{-1} \hat{\Sigma}_{\zeta\varphi}^{k\,T} \tag{3.42b}$$

$$r_k(\hat{\theta}_{k-1}) = \hat{\Sigma}_{\zeta\varphi}^k \hat{\theta}_{k-1} - \hat{\xi}_{\zeta y}^k \tag{3.42c}$$

For a time-invariant system, the stepsize (or gain sequence) is usually chosen as $\gamma_k = 1/k$ (see Ljung & Söderström 1983, Section 5.6), whilst alternative choices might allow the introduction of adaptivity within the input measurement variance computation (cf. also Remark 3.1).

Steepest gradient algorithm with line search

In (Linden, Vinsonneau & Burnham 2008), a steepest gradient algorithm of the form

$$\hat{\sigma}_{\tilde{u}}^k = \hat{\sigma}_{\tilde{u}}^{k-1} - \gamma_k \, {J_k^{(\sigma_{\tilde{u}})}}^T r_k(\hat{\theta}_{k-1}) \tag{3.43}$$

has been proposed to update the input measurement noise variance estimate. Note that this corresponds to a Newton method with $V_k'' = I$. Rather than using $\gamma_k = 1/k$ or choosing a constant step size as in (Linden, Vinsonneau & Burnham 2008), it is possible to perform a line search (cf. Dennis & Schnabel 1996, Section 6.3), in order to obtain an 'optimal' step size at each time instance k. This means that γ_k is chosen in a manner, such that the residual $r(\hat{\theta}_k)$ becomes minimal for the negative gradient update direction. For this purpose, it is again possible to make use of the linearised Frisch scheme equations (3.27). This is, however, equivalent to minimising an alternative cost function, which is solely based on the linearisations $L_\theta(\hat{\vartheta}_{k-1})$ and $L_{\sigma_{\tilde{y}}}(\hat{\vartheta}_{k-1})$ rather

than (3.1a) and (3.1b). Such an approach has been discussed in (Linden, Larkowski & Burnham 2008).

Minimisation of approximate cost function By linearising the Frisch equations around the latest estimate $\hat{\vartheta}_{k-1}$, an approximate cost function, which only depends on the linearised Frisch equations rather than (3.1a) and (3.1b), can be defined as

$$V_k^{\text{lin}} \triangleq \frac{1}{2} \left\| r_k \left(L_\theta(\hat{\vartheta}_{k-1}) \right) \right\|_2^2. \tag{3.44}$$

In order to determine that $\hat{\sigma}_{\tilde{u}}^k$ which minimises (3.44), the total derivative of V_k^{lin} with respect to $\hat{\sigma}_{\tilde{u}}^k$ is required. This is given by (Björck 1996, cf. p. 340)

$$\frac{dV_k^{\text{lin}}}{d\hat{\sigma}_{\tilde{u}}^k} = J^T(\hat{\sigma}_{\tilde{u}}^k) r_k \left(L_\theta(\hat{\vartheta}_{k-1}) \right), \tag{3.45}$$

where $J(\hat{\sigma}_{\tilde{u}}^k)$ denotes the Jacobian defined by (3.38). The dependencies of r_k, L_θ and $L_{\sigma_{\tilde{y}}}$ are dropped in the subsequent development for the ease of notation. Setting the total derivative (3.45) equal to zero and substituting (3.29) and (3.40) yields

$$\begin{aligned} 0 = J^T(\hat{\sigma}_{\tilde{u}}^k) \Big(\hat{\Sigma}_{\zeta\varphi}^k \big[\hat{\theta}_{k-1} + \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \iota(\hat{\vartheta}_{k-1}) \\ + \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \kappa(\hat{\vartheta}_{k-1}) \hat{\sigma}_{\tilde{u}}^k \big] - \hat{\xi}_{\zeta y}^k \Big), \end{aligned} \tag{3.46}$$

from which the input measurement noise variance is computed as

$$\hat{\sigma}_{\tilde{u}}^k = \frac{J^T(\hat{\sigma}_{\tilde{u}}^k) \left(\hat{\Sigma}_{\zeta\varphi}^k \left[\hat{\theta}_{k-1} + \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \iota(\hat{\vartheta}_{k-1}) \right] - \hat{\xi}_{\zeta y}^k \right)}{-J^T(\hat{\sigma}_{\tilde{u}}^k) \hat{\Sigma}_{\zeta\varphi}^k \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \kappa(\hat{\vartheta}_{k-1})}. \tag{3.47}$$

Note that the Jacobian is given in a straightforward manner from (3.40b) by

$$J(\hat{\sigma}_{\tilde{u}}^k) = \hat{\Sigma}_{\zeta\varphi}^k \frac{dL_\theta}{d\hat{\sigma}_{\tilde{u}}^k}, \tag{3.48}$$

whilst the total derivative of L_θ is obtained from (3.29) as

$$\frac{dL_\theta}{d\hat{\sigma}_{\tilde{u}}^k} = \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \kappa(\hat{\vartheta}_{k-1}). \tag{3.49}$$

Finally, substituting (3.49) into (3.48), the Jacobian becomes

$$J(\hat{\sigma}_{\tilde{u}}^k) = \hat{\Sigma}_{\zeta\varphi}^k \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \kappa(\hat{\vartheta}_{k-1}). \tag{3.50}$$

The algorithm, which minimises the YW cost function by making use of the linearised Frisch equations is denoted YW-lin and can be summarised as follows.

Algorithm 3.6 (YW-lin).

$$\hat{\sigma}_{\tilde{u}}^{k} = \frac{J^T(\hat{\sigma}_{\tilde{u}}^{k}) \left(\hat{\Sigma}_{\zeta\varphi}^{k} \left[\hat{\theta}_{k-1} + \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \iota(\hat{\vartheta}_{k-1}) \right] - \hat{\xi}_{\zeta y}^{k} \right)}{-J^T(\hat{\sigma}_{\tilde{u}}^{k}) \hat{\Sigma}_{\zeta\varphi}^{k} \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) \kappa(\hat{\vartheta}_{k-1})} \tag{3.51a}$$

$$\iota(\hat{\vartheta}_{k-1}) = \hat{\xi}_{\varphi y}^{k} - \hat{\Sigma}_{\varphi}^{k} \hat{\theta}_{k-1} + \left[\hat{\sigma}_{\tilde{y}}^{k-1} + \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\tilde{a}}_{k-1}^T \hat{\tilde{a}}_{k-1}} \hat{\sigma}_{\tilde{u}}^{k-1} \right] \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} \tag{3.51b}$$

$$\kappa(\hat{\vartheta}_{k-1}) = \begin{bmatrix} -\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\tilde{a}}_{k-1}^T \hat{\tilde{a}}_{k-1}} \hat{a}_{k-1}^T & \hat{b}_{k-1}^T \end{bmatrix}^T \tag{3.51c}$$

$$\Sigma_{\varphi_0}(\sigma_{k-1}) = \hat{\Sigma}_{\varphi}^{k} - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \tag{3.51d}$$

3.3.5 Summary of recursive Frisch scheme algorithms

Based on the two different algorithms for the computation of the input measurement noise variance, two distinct (but similar) algorithms for recursive Frisch scheme identification can be formulated. The first algorithm, which utilises the Gauss-Newton search for the determination of $\hat{\sigma}_{\tilde{u}}^{k}$ is termed RFSa and can be summarised as follows.

Algorithm 3.7 (RFSa).

1) Initialisation

2) For $k = n_\zeta + n_b + 1, ...$

- **a)** Update γ_k via (3.8) and $\hat{\Sigma}_{\bar{\varphi}}^{k}$, $\hat{\Sigma}_{\zeta\bar{\varphi}}^{k}$ via (3.5)
- **b)** Compute P_k and $\hat{\theta}_k^{\mathrm{LS}}$ using Algorithm 3.3
- **c)** Update $\hat{\sigma}_{\tilde{u}}^{k}$ via Algorithm 3.5
- **d)** Update $\hat{\sigma}_{\tilde{y}}^{k}$ by means of Algorithm 3.4
- **e)** Compute $\hat{\theta}_k$ via Algorithm 3.2

The second algorithm, which makes use of the YW-lin approach, is denoted RFSb and is, for completeness, summarised as follows.

Algorithm 3.8 (RFSb).

1) Initialisation

2) For $k = n_\zeta + n_b + 1, ...$

a) Update γ_k via (3.8) and $\hat{\Sigma}^k_{\bar{\varphi}}$, $\hat{\Sigma}^k_{\zeta\bar{\varphi}}$ via (3.5)

b) Compute P_k and $\hat{\theta}^{\mathrm{LS}}_k$ using Algorithm 3.3

c) Update $\hat{\sigma}^k_{\tilde{u}}$ via Algorithm 3.6

d) Update $\hat{\sigma}^k_{\tilde{y}}$ by means of Algorithm 3.4

e) Compute $\hat{\theta}_k$ via Algorithm 3.2

Remark 3.4 (Projection facility). In order to stabilise the recursive algorithms during the initial phase, it might be advantageous to project the noise variance estimates into the intervals

$$0 \leq \hat{\sigma}_{\tilde{u}} \leq \sigma^{\max}_{\tilde{u}}, \tag{3.52a}$$

$$0 \leq \hat{\sigma}_{\tilde{y}} \leq \sigma^{\max}_{\tilde{y}}, \tag{3.52b}$$

where $\sigma^{\max}_{\tilde{u}}$ and $\sigma^{\max}_{\tilde{y}}$ are the maximal admissible solutions for $\hat{\sigma}_{\tilde{u}}$ and $\hat{\sigma}_{\tilde{y}}$, respectively. Recall from Section 2.4.3 that these intervals are naturally motivated from the Frisch scheme approach and the maximal admissible values can be computed from the data as described in (2.48)

$$\sigma^{\max}_{\tilde{u}} = \lambda_{\min}\left[\hat{\Sigma}^k_{\varphi_u} - \hat{\Sigma}^k_{\varphi_u\varphi_y}[\hat{\Sigma}^k_{\varphi_y}]^{-1}\hat{\Sigma}^k_{\varphi_y\varphi_u}\right], \tag{3.53a}$$

$$\sigma^{\max}_{\tilde{y}} = \lambda_{\min}\left[\hat{\Sigma}^k_{\varphi_y} - \hat{\Sigma}^k_{\varphi_y\varphi_u}[\hat{\Sigma}^k_{\varphi_u}]^{-1}\hat{\Sigma}^k_{\varphi_u\varphi_y}\right]. \tag{3.53b}$$

Since these boundaries rely on the solution of two eigenproblems, it would be more pragmatic to consider positive constants for the maximum admissible values, if such a priori knowledge is available. For the cases where the estimates exceed these maximal admissible values, it seems reasonable to set $\hat{\sigma}^k_{\tilde{u}} = \hat{\sigma}^{k-1}_{\tilde{u}}$ and/or $\hat{\sigma}^k_{\tilde{y}} = \hat{\sigma}^{k-1}_{\tilde{y}}$, respectively.

Remark 3.5 (Computational complexity). The computation time per single recursion can be reduced by approximately two-thirds by making use of the recursive Frisch-YW approaches compared to the RAFS (cf. Algorithm 3.1), although both algorithms are of cubic complexity with respect to the number of system parameters to be identified (see Section 3.4 below). However, approximate fast algorithms of quadratic order are possible by accounting for the fact that the eigenvector corresponding to the smallest eigenvalue of A_k is also part of the parameter vector to be estimated. This is further discussed in Chapter 4.

Remark 3.6 (Classification of the RFS algorithms). The recursive Frisch-YW algorithms can be considered to belong to the family of iterative bias-compensating LS

algorithms (see e.g. (Sagara & Wada 1977, Zheng & Feng 1989, Söderström 2007*b*) and the references within). The essential distinguishing feature is, however, that the measurement noise variances are computed in a different way which is based on the offline Frisch scheme approach.

3.3.6 Numerical example

This first example aims to illustrate that the developed recursive Frisch scheme algorithms are able to work satisfactorily when applied to an arbitrarily simulated system, i.e. that the parameter estimates can successfully compensate for the bias. In addition, since the recursive algorithms are developed based on the offline Frisch scheme, it is interesting to investigate whether, and if so to quantify how much, the recursive schemes will deviate from the offline estimates, due to the introduced assumptions and approximations.

Remark 3.7. It should be noted that, in this example and others throughout the thesis, the RLS estimates have been utilised as a benchmark against which to evaluate the performance of the various recursive EIV algorithms developed in the present study. In practice, it is unlikely that the RLS algorithm would be used in such noisy situations, other than for the limited number of cases when auto-regressive with exogeneous inputs models are justified. To some extent, therefore, the use of RLS as a benchmark exaggerates the advantages of the EIV algorithms in practical terms, since superior optimal algorithms, such as the recursive prediction error minimisation algorithm in the Matlab System Identification Toolbox (Mathworks 2008) and the recursive refined instrumental variable algorithm in (*CAPTAIN Toolbox for Matlab* 2008), would be more appropriate. In this first example, for instance, the recursive prediction error minimisation algorithm and the recursive refined instrumental variable algorithm yield estimates that have very small asymptotic bias on the transfer function denominator estimates and quite small asymptotic bias on the transfer function numerator coefficients. As a result, the associated frequency response characteristics differ little from the actual ones (see Section 8.8 in Young, Taylor & Chotai 2009) and the resulting model would be quite acceptable for many practical control applications. Of course, these biases would be larger for higher noise levels and so, in noisy situations where parametric bias is particularly important, the recursive EIV algorithms provide a superior approach to data-based modelling and it is for such applications as these that the algorithms described in this thesis have been developed.

Consider a LTI SISO system with $n_a = n_b = 2$, which is given by (cf. (2.4a) and (2.9))

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix}^T, \tag{3.54a}$$

$$\sigma = \begin{bmatrix} 2.1 & 0.1 \end{bmatrix}^T. \tag{3.54b}$$

The input and output measurement noise variances are chosen, such that an equal signal-to-noise ratio of 10dB is obtained at the input as well as the output of the EIV system. The system is simulated for 500 samples using a zero mean, white and Gaussian distributed input signal of unity variance. The RFSa and the RFSb are applied to estimate ϑ; both using $n_\zeta = n_a + n_b + 1$ as the number of instruments for the YW model selection criterion. In order to compare the recursive estimates with those obtained by the offline Frisch-YW, the RAFS is applied[6]. Since the system is time invariant, no adaptivity, i.e. $\lambda = 1$, is considered. The input and output measurement noise variances are projected into the intervals $[0, \sigma_{\tilde{u}}^{\max}]$ and $[0, \sigma_{\tilde{y}}^{\max}]$, respectively. The maximal admissible values for the input and output measurement noise variances are chosen to be $\sigma_{\tilde{u}}^{\max} = 2\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}^{\max} = 2\sigma_{\tilde{y}}$, respectively. The resulting estimates of ϑ are shown in Figure 3.1.

The dashed line corresponds to the offline estimate using 500 samples. As expected, the RAFS yields identical estimates for $k = 500$ recursions, whilst the estimates of the RFSa and RFSb are slightly different. Considering the estimates of θ, it is observed that the RFSa and RFSb can successfully compensate for the bias in the estimates, compared to the uncompensated RLS estimates. The most significant difference between the recursive estimates and the RAFS estimates is observed for $\sigma_{\tilde{u}}$. The Gauss-Newton estimate of the RFSa is much smoother than the RFSb estimate obtained via the minimisation of the modified YW cost function. This is an expected result, since the former weights new data with $1/k$, whilst the latter performs a line search at each time instance, which can result in rapid changes for the estimate of $\sigma_{\tilde{u}}$. For the RFS estimates of $\sigma_{\tilde{u}}$ it is also observed, that the projection facility is active for the first 200 recursions, which results in the 'flat' periods within this interval. Apart from this, the RFSb is able to approximate the offline estimate for $\sigma_{\tilde{u}}$ surprisingly well and it seems to be superior than that obtained by the RFSa. The estimates for $\sigma_{\tilde{y}}$ are virtually identical in all three cases, which indicates that the conjugate gradient method appears to satisfactorily to track the smallest eigenvalue of $\hat{A}_k$.

It is also of interest to compare the YW cost function V_k with its approximation V_k^{lin}, which has been used for the determination of $\sigma_{\tilde{u}}$ in the case of RFSb. Both are shown for $k = 80$, $k = 500$ and $k = 2500$ samples in Figure 3.2. It is observed that after 80 recursion steps, $\sigma_{\tilde{u}}$ corresponding to the minimum of V_k^{lin} (around $\sigma_{\tilde{u}} = 0.25$)

[6]The number of iterations for the minimisation of the YW cost function is not restricted to unity within this example, in order to obtain the exact offline estimates at each time instance k.

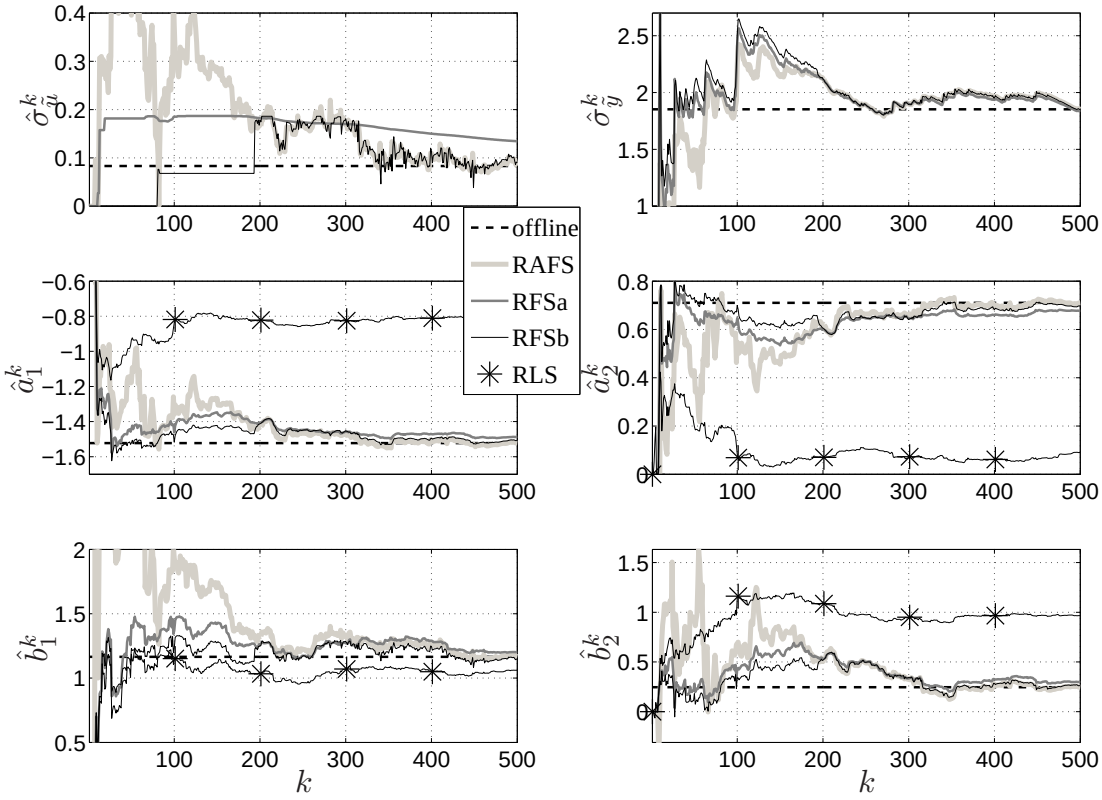


Figure 3.1: Estimates of ϑ_k for system (3.54) using the offline Frisch scheme, RAFS, RFSa, RFSb and RLS.

is already close to $\sigma_{\tilde{u}}$ corresponding to the minimum of V_k (around $\sigma_{\tilde{u}} = 0.5$). After 500 recursion steps, the difference between both minima is less than $8 \cdot 10^{-3}$ and for $k = 2500$, both minima virtually coincide. These results are in alignment with the good estimation performance for $\sigma_{\tilde{u}}$ in the case of RFSb, which has been observed in Figure 3.1.

Concerning Remark 3.3, it is interesting to investigate how accurate the computation of $\hat{\sigma}_{\tilde{y}}^k$ is, i.e. how exact the least eigenvalue of $\hat{A}_k$ can be approximated using the conjugate gradient method. Therefore, the experiment is repeated where the RFSa[7] utilises the same input measurement noise variance estimate as the exact algorithm, i.e. $\hat{\sigma}_{\tilde{u}}^k$ is identical in both cases. The differences in $\hat{\sigma}_{\tilde{y}}^k$ is then a measure for the accuracy of the subspace tracking algorithm. It turns out that the difference between both estimates is marginal: after 500 recursion steps, for instance, it is approximately $6 \cdot 10^{-7}$ while it decreases to $8 \cdot 10^{-8}$ after 5000 iterations. This means (at least in the example considered here) that the RFS algorithms yield estimates of $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$ which seem to converge to the set of admissible Frisch solutions (cf. Section 2.4.3), once the initialisation transients have decayed. This aspect is further investigated in Section 3.5.

[7]Since both RFSa and RFSb use the CG-GN algorithm, here, it is irrelevant which algorithm is chosen.

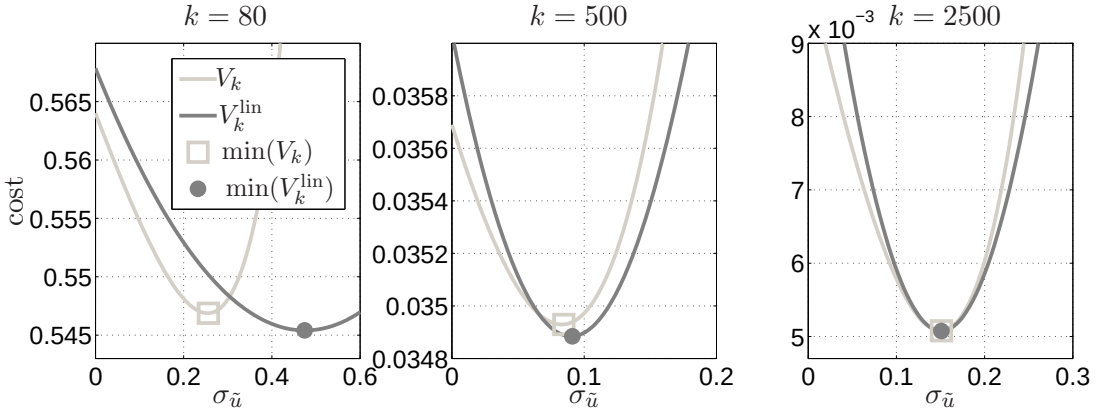


Figure 3.2: Comparison of cost function values for V_k (3.22b) and V_k^{lin} (3.44) after $k = 80$, $k = 500$ and $k = 2500$ recursions.

3.4 Computational complexity

A recursive identification scheme is usually intended for online usage, i.e. to obtain estimates while the process under consideration is in operation. Consequently, it is important to analyse the computational complexity of the scheme. If, for example, the parameters are estimated at each sample, the computation time required per recursion is restricted by the choice of sampling interval for the process as well as the utilised hardware. Hence the analysis of the computational complexity of the RFS algorithms is an important issue with respect to their applicability in practical situations.

This section provides a detailed analysis of the computational complexity of the developed RFSa and RFSb algorithms and compares this with the RAFS algorithm (which is essentially a repeated application of the offline Frisch scheme equations).

3.4.1 RAFS algorithm

A detailed description of the RAFS algorithm together with its computational costs (cf. Section 2.3.3) is given in Table 3.1. Steps 1 and 2 are initialisations of the algorithm which are not taken into account, since it is the computational complexity per recursion (steps inside the loop) that is of interest. The optimisation for the determination of the 'optimal' estimate for the input measurement noise variance in Step 3.4 (YW model selection) iterates only once per recursion, which is done in order to achieve a fixed number of RAFS flops. If `fminsearch` (Nelder-Mead simplex method) is used within a Matlab implementation (MathWorks 2007), this would correspond to setting the number of maximal iterations to unity.

It is clear that an application of the offline Frisch-YW equations at each time step k is rather crude and the overall costs are of cubic order with respect to the number of model parameters n_θ as observed in Table 3.1.

Step	Description	Procedure	Flops
1	Choose n_ζ, λ_k and j	$n_\zeta = n_a + n_b + 1$, $0 < \lambda_k < 1$, $j = n_\zeta + n_b$	
2	Initialise	$\hat{\sigma}_u^j = 0$, $\hat{\theta}_j = 0$, $\hat{\Sigma}_{\zeta\bar{\varphi}}^j = 0$, $\hat{\Sigma}_{\bar{\varphi}}^j = \frac{1}{n_\zeta - 1}\sum_{i=n_b+1}^{j} \bar{\varphi}_i \bar{\varphi}_i^T$ and $\gamma_j = 1/(n_\zeta - 1)$	
3	Recursion	for $k = j+1, ...$	
3.1	Update γ	$\gamma_k = \frac{\gamma_{k-1}}{\lambda_k + \gamma_{k-1}}$	2
3.2	Update $\hat{\Sigma}_{\bar{\varphi}}$	$\hat{\Sigma}_{\bar{\varphi}}^k = \hat{\Sigma}_{\bar{\varphi}}^{k-1} + \gamma_k \left(\bar{\varphi}_k \bar{\varphi}_k^T - \hat{\Sigma}_{\bar{\varphi}}^{k-1} \right)$	$O(n_{\bar{\theta}}^2)$
3.3	Update $\hat{\Sigma}_{\zeta\bar{\varphi}}$	$\hat{\Sigma}_{\zeta\bar{\varphi}}^k = \hat{\Sigma}_{\zeta\bar{\varphi}}^{k-1} + \gamma_k \left(\zeta_k \bar{\varphi}_k^T - \hat{\Sigma}_{\zeta\bar{\varphi}}^{k-1} \right)$	$O(n_{\bar{\theta}}^2)$
3.4	Update $\hat{\sigma}_u$	$\hat{\sigma}_u^k = \arg\min_{\sigma_u} \|\hat{\Sigma}_{\zeta\bar{\varphi}}^k \bar{\theta}\|_2^2$ (via optimisation, costs given per single iteration)	$O(n_\theta^3)$
3.5	Update $\hat{\sigma}_y$	$\hat{\sigma}_y^k = \lambda_{\min}(\hat{A}_k)$	$O(n_{\bar{a}}^3 + n_{\bar{a}}^2 n_b + n_b^3)$
3.6	Update $\hat{\theta}$	solve $\left(\hat{\Sigma}_{\varphi}^k - \Sigma_{\bar{\varphi}}(\hat{\sigma}_k) \right) \hat{\theta}_k = \hat{\xi}_{\varphi y}$	$O(n_\theta^3)$
		Overall complexity (dominant parts)	$O(n_\theta^3)$

Table 3.1: Computational complexity of the repeatedly applied Frisch scheme (RAFS) algorithm.

Step	Procedure	Flops
1	$L_k = \frac{P_{k-1}\varphi_k}{\varphi_k^T P_{k-1} \varphi_k + \frac{1-\gamma_k}{\gamma_k}}$	$O(n_\theta^2)$
2	$\hat{\theta}_k^{\mathrm{LS}} = \hat{\theta}_{k-1}^{\mathrm{LS}} + L_k \left(y_k - \varphi_k^T \hat{\theta}_{k-1}^{\mathrm{LS}} \right)$	$O(n_\theta)$
3	$P_k = \frac{1}{1-\gamma_k} \left(P_{k-1} - \frac{P_{k-1}\varphi_k \varphi_k^T P_{k-1}}{\varphi_k^T P_{k-1} \varphi_k + \frac{1-\gamma_k}{\gamma_k}} \right)$	$O(n_\theta^2)$
	Overall complexity (dominant parts)	$O(n_\theta^2)$

Table 3.2: Computational complexity of RLS using scaled P_k.

3.4.2 RFS algorithms

This subsection provides a detailed listing of the computational costs of the RFSa and RFSb algorithms.

Recursive least squares

The RFS algorithms are essentially recursive BCLS procedures based on the RLS algorithm. A basic RLS form together with its computational complexity is given in Table 3.2. It is observed that the RLS is of quadratic complexity with respect to the number of model parameters n_θ. Note that there exist numerically more robust and efficient RLS implementations other than the version presented here (Sayed & Kailath 1994*a*). For the analysis in this section, however, this is irrelevant since the bottlenecks of the RFS algorithms lie elsewhere as pointed out below.

YW-GN algorithm

In order to track the minimum of the YW cost function (3.22b), a Gauss-Newton algorithm has been proposed (Algorithm 3.5). The computational complexity of the YW-GN algorithm is listed in Table 3.3. It is observed that the bottlenecks for the

Step	Description	Procedure	Flops
1	Derivative	$\theta_k^{(\tilde{u})} = \left(\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right)^{-1} \begin{bmatrix} 0 \\ \hat{b}_{k-1} \end{bmatrix}$	$O(n_\theta^3)$
2	Derivative	$\theta_k^{(\tilde{y})} = \left(\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right)^{-1} \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix}$	$O(n_\theta^3)$
3	Derivative	$\sigma_{\tilde{y},k}^{(\tilde{y})} = -(\hat{b}_{k-1}^T \hat{b}_{k-1})/(\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1})$	$O(n_b + n_a)$
4	Jacobian	$J_k^{(\sigma_{\tilde{u}})^T} = \hat{\Sigma}_{\zeta\varphi}^k \left(\theta_k^{(\tilde{y})} \sigma_{\tilde{y},k}^{(\tilde{u})} + \theta_k^{(\tilde{u})}\right)$	$O(n_\theta)$
5	Update $\hat{\sigma}_{\tilde{u}}$	$\hat{\sigma}_{\tilde{u}}^k = \hat{\sigma}_{\tilde{u}}^{k-1} + \gamma_k \left[J_k^{(\sigma_{\tilde{u}})^T} J_k^{(\sigma_{\tilde{u}})}\right]^{-1} J_k^{(\sigma_{\tilde{u}})^T} r_k(\hat{\theta}_{k-1})$	$O(n_\theta)$
		Overall complexity (dominant parts)	$O(n_\theta^3)$

Table 3.3: Computational complexity of YW-GN algorithm.

Step	Description	Procedure	Flops
1	Update $\sigma_{\tilde{u}}$	$\hat{\sigma}_{\tilde{u}}^k = \frac{J^T(\hat{\sigma}_{\tilde{u}}^k)\left(\hat{\Sigma}_{\zeta\varphi}^k \left[\hat{\theta}_{k-1} + \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1})\iota(\hat{\vartheta}_{k-1})\right] - \hat{\xi}_{\zeta y}^k\right)}{-J^T(\hat{\sigma}_{\tilde{u}}^k)\hat{\Sigma}_{\zeta\varphi}^k \Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1})\kappa(\hat{\vartheta}_{k-1})}$	$O(n_\theta^3)$
2		$\iota(\hat{\vartheta}_{k-1}) = \hat{\xi}_{\varphi y}^k - \hat{\Sigma}_\varphi^k \hat{\theta}_{k-1} + \left[\hat{\sigma}_{\tilde{y}}^{k-1} + \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{\sigma}_{\tilde{u}}^{k-1}\right] [\hat{a}_{k-1}^T \; 0]^T$	$O(n_\theta^2)$
3		$\kappa(\hat{\vartheta}_{k-1}) = \begin{bmatrix} -\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{a}_{k-1}^T & \hat{b}_{k-1}^T \end{bmatrix}^T$	$O(n_{\bar{a}})$
4		$\Sigma_{\varphi_0}(\sigma_{k-1}) = \hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})$	n_θ
		Overall complexity (dominant parts)	$O(n_\theta^3)$

Table 3.4: Computational complexity of the YW-lin algorithm.

Gauss-Newton method are clearly Steps 1 and 2, i.e. the computation of the derivatives $\theta_k^{(\tilde{u})}$ and $\theta_k^{(\tilde{y})}$ due to the matrix inversion that is involved being of cubic complexity.

YW-lin algorithm

The YW-lin algorithm which minimises the modified cost function (3.44) is summarised in Table 3.4. It is observed here that the bottleneck is, as in the case of YW-GN, the computation of the matrix inverse $\Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) = \left(\hat{\Sigma}_\varphi^k - \hat{\Sigma}_{\tilde{\varphi}}^{k-1}\right)^{-1}$, which is of cubic complexity. Therefore, the overall complexity of the YW-lin algorithm is also $O(n_\theta^3)$.

Conjugate gradient method

For the computation of the output measurement noise variance the least eigenvalue of the matrix $\hat{A}_k$ in (3.2b) is required to be determined. Whilst a complete eigenvalue decomposition generally requires $O(n^3)$ flops with n being the size of the square matrix, here only the smallest eigenvalue is of interest. For this purpose, the conjugate gradient subspace tracking algorithm has been proposed. The procedure, which is of $O(n^2)$

Step	Description	Procedure	Flops
1	Schur complement	$\hat{A}_k = \hat{\Sigma}^k_{\bar{\varphi}_y} - \hat{\Sigma}^k_{\bar{\varphi}_y\varphi_u}\left[\hat{\Sigma}^k_{\varphi_u} - \hat{\sigma}^k_u I_{n_b}\right]^{-1}\hat{\Sigma}^k_{\varphi_u\bar{\varphi}_y}$	$O(n_b^3 + n_{\bar{a}}^2 n_b + n_{\bar{a}} n_b^2)$
2	Eigenvector	$\hat{x}_k = \hat{\bar{x}}_{k-1} + \hat{\mu}_k\hat{\psi}_{k-1}$	$O(n_{\bar{a}})$
3	Normalisation	$\hat{\bar{x}}_k = \hat{x}_k / \left(\hat{x}_k^T\hat{x}_k\right)^{1/2}$	$O(n_{\bar{a}})$
4	Update $\hat{\sigma}_y$	$\hat{\sigma}^k_y = \hat{\bar{x}}_k^T\hat{A}_k\hat{\bar{x}}_k$	$O(n_{\bar{a}}^2)$
5	Residual	$\hat{r}_k = \hat{A}_k\hat{\bar{x}}_k - \hat{\sigma}^k_{\hat{y}}\hat{\bar{x}}_k$	$O(n_{\bar{a}}^2)$
6		$\hat{q}_{k-1} = -(\hat{r}_k^T\hat{A}_k\hat{\psi}_{k-1})/(\hat{\psi}_{k-1}^T\hat{A}_k\hat{\psi}_{k-1})$	$O(n_{\bar{a}}^2)$
7	Update direction	$\hat{\psi}_k = \hat{r}_k + \hat{q}_{k-1}\hat{\psi}_{k-1}$	$O(n_{\bar{a}})$
8		$d_1^k = \hat{\bar{x}}_k^T\hat{A}_k\hat{\psi}_k$	$O(n_{\bar{a}}^2)$
9		$d_2^k = \hat{\psi}_k^T\hat{A}_k\hat{\psi}_k$	$O(n_{\bar{a}}^2)$
10		$d_3^k = \hat{\bar{x}}_k^T\hat{\psi}_k$	$O(n_{\bar{a}})$
11		$d_4^k = \hat{\psi}_k^T\hat{\psi}_k$	$O(n_{\bar{a}})$
12		$d_5^k = \left(\hat{\sigma}^k_{\hat{y}}d_4^k - d_2^k\right)^2 - 4\left(d_2^k d_3^k - d_1^k d_4^k\right)$ $\times\left(d_1^k - \hat{\sigma}^k_{\hat{y}}d_3^k\right)$	10
13	Optimal step size	$\hat{\mu}_k = \left(\hat{\sigma}^k_{\hat{y}}d_4^k - d_2^k + \sqrt{d_5^k}\right) / \left(2\left(d_2^k d_3^k - d_1^k d_4^k\right)\right)$	16
		Overall complexity (dominant parts)	$O(n_b^3 + n_{\bar{a}}^2 n_b + n_{\bar{a}} n_b^2)$

Table 3.5: Conjugate gradient method for tracking smallest eigenvalue σ_y (CG-RQ algorithm).

complexity[8], is summarised in Table 3.5. Although the conjugate gradient method is, in general, only of quadratic order, in this particular application $O(n_b^3)$ flops are necessary. Upon examination of Step 1 in Table 3.5 it becomes clear, that the computation of the Schur complement of the block matrix $\hat{\Sigma}^k_{\varphi_u} - \hat{\sigma}^k_u I_{n_b}$ is the bottleneck within the conjugate gradient method. The matrix inversion in (3.2b) requires $O(n_b^3)$ and the matrix multiplications require another $O(n_{\bar{a}}^2 n_b + n_{\bar{a}} n_b^2)$ flops.

Overall complexity

A detailed description as well as the computational costs of the RFS algorithms are given in Table 3.6. Since only the computation of the input measurement noise variance differentiates the RFSa algorithm from the RFSb algorithm, Table 3.6 encompasses both cases (cf. Step 6.5). Steps 1 to 5 are initialisations for the algorithm which are not taken into account since it is the computational complexity per recursion which is of interest. The remaining steps summarise the computational complexity of the parts of the algorithm which have been discussed in detail in the previous paragraphs. As in the case of RAFS, the overall computational complexity of the RFS algorithms is of cubic order with respect to n_θ, the number of model parameters to be identified. This would apparently represent an undesirable feature, since it also raises questions regarding the effort which has been expended in order to derive a fully recursive version of the

[8]Linear complexity for the eigenvalue tracking problem can only be achieved for a rank-one update of the corresponding matrix. However, since the update $\Delta A_k = A_k - A_{k-1}$ is generally of full rank, $O(n^2)$ flops are required.

Step	Description	Procedure	Flops
1	Choose n_ζ, λ_k and j	$n_\zeta = n_a + n_b + 1$, $0 < \lambda_k < 1$, $j = n_\zeta + n_b$	
2	RLS initialisations	$\hat{\theta}^{\mathrm{LS}}_{n_a} = 0$, $P_{n_a} = 0.1I$	
3	Recursion	for $k = n_a + 1, ..., j$	
3.1	Data weighting	$\gamma_k = 1/k$	
3.2	RLS	Compute L_k, $\hat{\theta}^{\mathrm{LS}}_k$ and P_k as in Table 3.2	
4	CG initialisations	Obtain 'guess' for eigenvector x_j associated with smallest eigenvalue	
4.1	Normalised eigenvector	$\bar{x}_j = x_j \left(x_j^T x_j\right)^{1/2}$	
4.2	Update direction	$\psi_j^x = \hat{A}_j \bar{x}_j - \left(\bar{x}_j^T \hat{A}_j \bar{x}_j\right) \bar{x}_j$	
5	General initialisations	$\hat{\Sigma}^j_{\zeta\bar{\varphi}} = 0$, $\hat{\Sigma}^j_{\bar{\varphi}} = \frac{1}{n_\zeta - 1} \sum_{i=n_b+1}^{j} \bar{\varphi}_i \bar{\varphi}_i^T$ and $\gamma_j = 1/(n_\zeta - 1)$	
6	Recursion	for $k = j + 1, ...$	
6.1	Update γ	$\gamma_k = \frac{\gamma_{k-1}}{\lambda_k + \gamma_{k-1}}$	2
6.2	Update $\hat{\Sigma}_{\bar{\varphi}}$	$\hat{\Sigma}^k_{\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\bar{\varphi}} + \gamma_k \left(\bar{\varphi}_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\bar{\varphi}}\right)$	$O(n_\theta^2)$
6.3	Update $\hat{\Sigma}_{\zeta\bar{\varphi}}$	$\hat{\Sigma}^k_{\zeta\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}} + \gamma_k \left(\zeta_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}}\right)$	$O(n_\theta^2)$
6.4	RLS estimate	Compute L_k, $\hat{\theta}^{\mathrm{LS}}_k$ and P_k as in Table 3.2	$O(n_\theta^2)$
6.5	Computation of $\hat{\sigma}^k_{\tilde{u}}$		
	either:	Compute $\hat{\sigma}^k_u$ as in Table 3.3	$O(n_\theta^3)$
	or:	Compute $\hat{\sigma}^k_u$ as in Table 3.4	$O(n_\theta^3)$
6.6	Projection	Project $0 \leq \hat{\sigma}^k_{\tilde{u}} \leq \sigma^{\max}_{\tilde{u}}$	
6.7	CG method for $\hat{\sigma}_y$	Compute $\hat{\sigma}^k_y$ as in Table 3.5	$O(n_b^3 + n_{\bar{a}}^2 n_b + n_{\bar{a}} n_b^2)$
6.8	Projection	Project $0 \geq \hat{\sigma}^k_{\tilde{y}} \geq \sigma^{\max}_{\tilde{y}}$	
6.9	Bias compensation	$\hat{\theta}_k = \hat{\theta}^{\mathrm{LS}}_k + P_k \hat{\Sigma}^k_{\bar{\varphi}} \hat{\theta}_{k-1}$	$O(n_\theta^2)$
		Overall complexity (dominant parts)	$O(n_\theta^3)$

Table 3.6: Computational complexity of gradient-based recursive Frisch-YW (RFSa and RFSb) algorithms. Abbreviation CG denotes conjugate gradient.

Frisch-YW algorithm. Consequently, this aspect requires some further investigation and analysis which is given in the following subsection.

3.4.3 Computation time comparison

This section compares the computation time per recursion of the RAFS and RFS algorithms in simulation. The algorithms are applied to a sequence of systems with an increasing order. To realise such a comparison, the model order, denoted here for convenience by $n \triangleq n_a = n_b$, of the system to be identified is increased incrementally from 1 to 30. Apart from the system, which is generating the data, a similar setup as in Section 3.3.6 is utilised. In particular, the signal-to-noise ratio is set to 10dB on both input and the output, respectively. In addition, the minimum number of instruments is chosen, i.e. $n_\zeta = 2n + 1$. The computation time for each recursion step using a certain model order is recorded. For each n, the minimum time for 500 recursions is

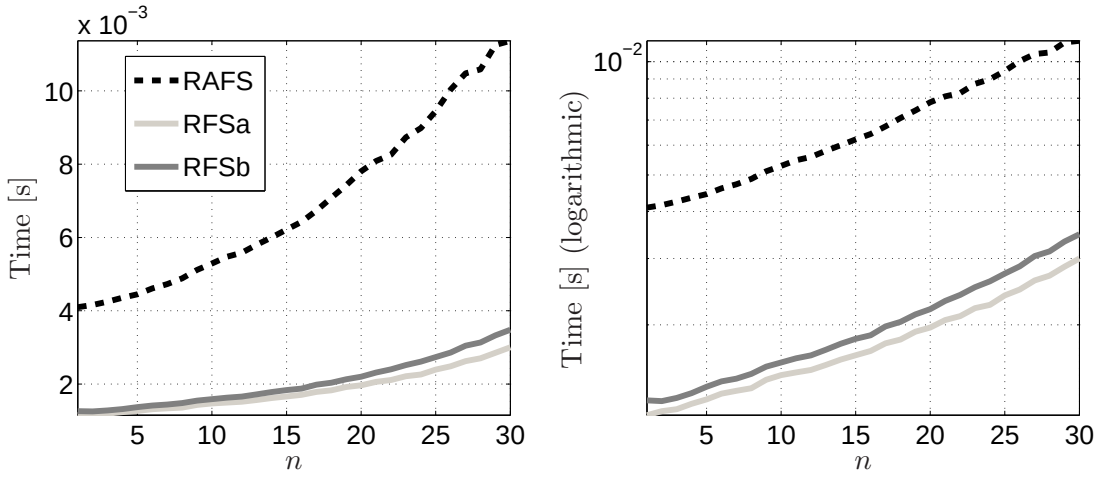


Figure 3.3: Computation time per single recursion with increasing model order n (linear and logarithmic scale).

considered, which helps to reduce the influencing effects of other applications running in the background of the operating system. In addition, for the RAFS the number of iterations for the optimisation of Step 3.4 in Table 3.1 is restricted to one. This is to ensure a meaningful comparison with the RFS algorithms[9]. The experiment has been carried out in MATLAB using the Linux operating system Ubuntu, whilst the hardware consists of an IBM ThinkCentre with Intel Pentium 4 HT processor having 3.6 GHz. The results are shown in Figure 3.3.

It is observed that the computation time per recursion of the RAFS algorithm, which basically applies the offline equations at each time step k, is greater than the time required by the RFS algorithms. In fact, for a model order of $n = 30$, the RFSa and RFSb approximately require less than one-third ($\approx$ 3ms and $\approx$ 3.5ms, respectively) of the computation time of the RAFS ($\approx$ 11ms). Comparing the RFSa and RFSb, the former appears to be slightly faster. The fact that the slopes of the curves corresponding to all three algorithms is similar would indicate that the computational complexity is of a similar order in all cases. Indeed, this is in agreement with the theoretical results obtained in this section; namely that all three algorithms are of cubic complexity. Although the RAFS and the RFS algorithms are of cubic complexity, the computation time can be significantly reduced by utilising the latter gradient-based recursive algorithms. The overall complexity of the RFS algorithms can be further reduced towards quadratic order by introducing various approximations. The resulting novel algorithms are derived and analysed in Chapter 4.

[9]In fact, to run one iteration as a new data set arrives is a natural step within recursive schemes. However, in this case, the RAFS will not yield identical results with respect to the offline algorithm, which is the reason why the number of iterations was not restricted in the previous simulation.

3.5 Frisch-character of recursive estimates

In contrast to other bias-compensating approaches, the solution of the Frisch scheme is uniquely characterised by the set of admissible solutions given by the convex curve in the noise space as illustrated in Figure 2.2. This curve is defined by (2.39b) and ensures that given an input measurement noise variance σ_u, the output measurement noise variance σ_y is selected such that the data becomes compatible with the chosen EIV model structure. As outlined in Remark 3.3, however, this Frisch-character (cf. Definition 2.1) of the solution holds only in an approximate sense when the RFS algorithms are applied. The reason for this is that in the recursive scheme, equation (3.15) is replaced by the subspace tracking algorithm given in (3.15)-(3.19), which can only approximate the smallest eigenvalue of $\hat{A}_k$. In addition, the error in $\hat{\sigma}_{\tilde{y}}^k$ is propagated to $\hat{\theta}_k$ due to (3.10) and, as a consequence, will subsequently affect $\hat{\sigma}_{\tilde{u}}^{k+1}$. This means that the point $(\hat{\sigma}_{\tilde{u}}^k, \hat{\sigma}_{\tilde{y}}^k)$ can no longer be guaranteed to lie on the convex curve in the noise space. Section 3.3.6 has briefly investigated this issue by observing the difference between $\sigma_{\tilde{y}}$ computed by (2.46b) and the conjugate gradient method for a particular example. This section investigates in somewhat more depth the question of how well the RFS schemes can retain the Frisch-character of the solution and, furthermore, how this can be measured online. Note that the interest in the Frisch-character of the solution is rather of an academic nature and might be considered as being of secondary importance for a practical application. However, it provides additional insight of how well the recursive algorithms are able to approximate the offline Frisch scheme.

Consider the following example.

Example 3.1. A LTI SISO EIV system is given by (cf. (2.4a) and (2.9))

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix} \tag{3.55a}$$

$$\sigma = \begin{bmatrix} 2 & 1 \end{bmatrix}^T \tag{3.55b}$$

with $n_a = n_b = 2$, whilst n_ζ is set to $n_{\bar{\theta}}$. For this setup, the signal-to-noise ratio for the input and output is given by 0.1dB and 23.1dB, respectively. The search for the input and output noise variances is restricted to the interval $0 \leq \hat{\sigma}_{\tilde{u}}^k \leq 2\sigma_{\tilde{u}}$ and $0 \leq \hat{\sigma}_{\tilde{y}}^k \leq 2\sigma_{\tilde{y}}$, respectively, which helps to stabilise the recursive schemes during the initialisation phase. Since a LTI system is considered, the forgetting factor is chosen to be $\lambda_k = 1$ for all k. The system is simulated for 100 samples and the RAFS and RFSb are applied to estimate ϑ recursively[10]. In addition, at each time instant k, the convex curve in the noise space is computed using (2.46b). The set of convex curves generating a hypersurface in $\mathbb{R}^3$ as well as $(\hat{\sigma}_{\tilde{u}}^k, \hat{\sigma}_{\tilde{y}}^k)$ for both RAFS and RFSb are shown in Figure 3.4. It is observed that the hypersurface is 'erratic' during the initial phase of the experiment, which is due to finite sample effects. As expected,

[10] Since the Frisch-character depends on the computation of $\hat{\sigma}_{\tilde{y}}^k$, only one RFS algorithm is considered.

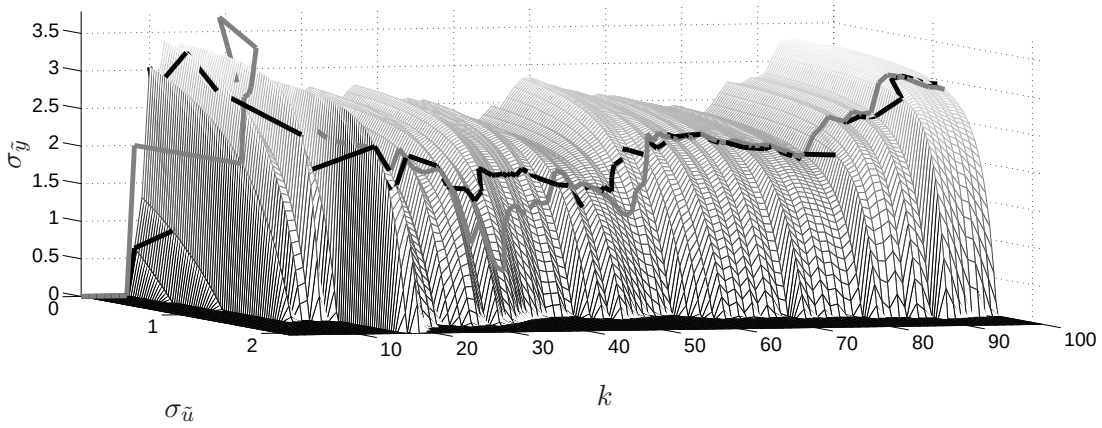


Figure 3.4: Convex curve in the noise space evolving with k discrete-time steps. The black solid line corresponds to the estimates $(\hat{\sigma}^k_{\tilde{u}}, \hat{\sigma}^k_{\tilde{y}})$ of the RAFS, whereas the grey line corresponds to the RFSb algorithm.

with an increasing number of samples the surface becomes smoothed since the effect of single samples becomes less significant. For the estimates of $(\sigma^k_{\tilde{u}}, \sigma^k_{\tilde{y}})$, it is observed, that whilst the black line corresponding to the noise variance estimates computed by the RAFS lies exactly on the hypersurface, the estimates obtained from the RFSb algorithm are considered as providing an approximation only. This observation is more evident during the first 20 runs of the algorithm, where the 'peaks' of the grey line corresponding to the RFSb, see Figure 3.4, depart from the hypersurface. ■

Example 3.1 has shown the a loss of the Frisch-character in the case of the recursive Frisch scheme. However, it is not evident from Figure 3.4 how close the estimates of the RFSb algorithm approximate the hypersurface with an evolving number of samples. In order to make a quantitative statement, appropriate measures need to be introduced. Note that satisfying the Frisch-character of the estimates basically depends on how accurate the smallest eigenvalue of $\hat{A}_k$ can be computed. One possible measure could be based on the difference between the 'exactly' and 'approximately' determined output measurement noise, as has been utilised in Section 3.3.6. This measure is denoted F^k_1 and is given by

$$F^k_1 \triangleq \left(\hat{\sigma}^{k*}_{\tilde{y}} - \hat{\sigma}^k_{\tilde{y}}\right)^2, \tag{3.56}$$

where $\hat{\sigma}^{k*}_y$ is computed via (2.46b) at each time instant k (as in the RAFS) whereas $\hat{\sigma}^k_{\tilde{y}}$ is obtained by the conjugate gradient method. This corresponds to the squared vertical distance between the point $(\hat{\sigma}^k_{\tilde{u}}, \hat{\sigma}^k_{\tilde{y}})$ obtained from the RFSb and the hypersurface. A more general possibility could be to compute the smallest eigenvalue of

$$\hat{\Sigma}^k_{\bar{\varphi}_0}(\hat{\sigma}_k) = \hat{\Sigma}^k_{\bar{\varphi}} - \begin{bmatrix} \hat{\sigma}^k_{\tilde{y}} I_{n_a+1} & 0 \\ 0 & \hat{\sigma}^k_{\tilde{u}} I_{n_b} \end{bmatrix}, \tag{3.57}$$

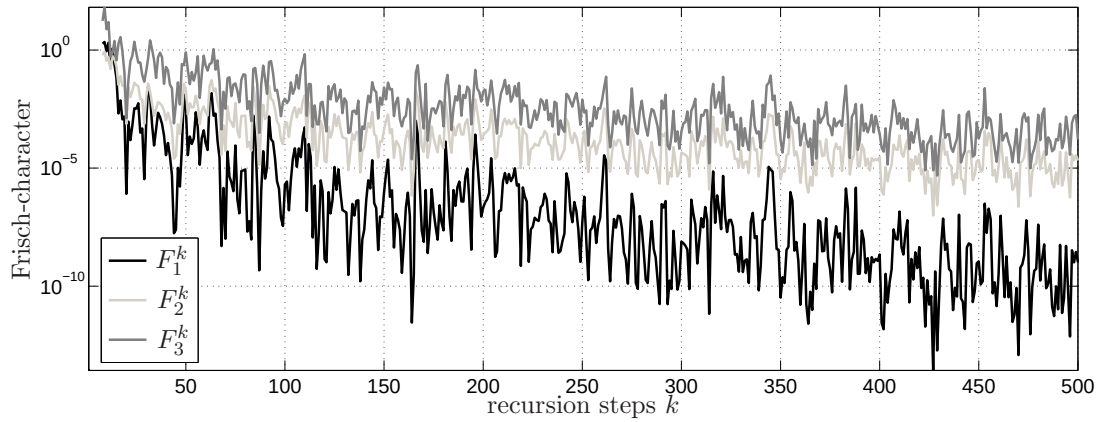


Figure 3.5: Different measures for assessing the Frisch-character of the RFS solutions.

which yields

$$F_2^k \triangleq \lambda_{\min}\left(\hat{\Sigma}^k_{\bar{\varphi}_0}(\hat{\sigma}_k)\right). \tag{3.58}$$

Recall that the aim of the Frisch scheme is to choose $\sigma^k_{\tilde{y}}$, such that

$$\hat{\Sigma}^k_{\bar{\varphi}_0}(\hat{\sigma}_k)\bar{\theta}_k = 0 \tag{3.59}$$

holds, i.e. that $\hat{\Sigma}^k_{\bar{\varphi}_0}$ is singular positive semidefinite, which means that its smallest eigenvalue is equal to zero in the exact case.

However, the measures F_1^k and F_2^k are somewhat of an academic nature since the exact eigenvalue computation is aimed to be avoided within the recursive algorithms. Recall that an eigenvalue decomposition is, in general, of cubic complexity. Hence, the use of these measures within the recursive algorithm appears to be impractical for monitoring the Frisch-character online due to its computational costs. A more suitable possibility could be to monitor the quantity $\hat{r}_k$ in (3.16a), since the residual provides information about the 'quality' of the conjugate gradient algorithm. Hence, the residual contains information about the Frisch-character of the RFS estimates and a third measure may be given by

$$F_3^k \triangleq ||\hat{r}_k||_2^2, \tag{3.60}$$

which can easily be computed online. The following example compares these measures in simulation.

Example 3.2. Consider the system of Example 3.1 using 500 recursions. All three performance measures for the Frisch-character are shown in Figure 3.5. It is observed that all measures exhibit a similar trend and tend towards zero with evolving k. The value for F_1^k, i.e. the squared error between the exactly computed eigenvalue and that

computed using the conjugate gradient method, is around 10^{-10} after 500 recursions. In addition, F_2^k and F_3^k are around $10 \cdot 10^{-4}$ and appear to be very similar. ∎

Since the 'cheaply' computed measure F_3^k is very similar to F_2^k and exhibits a similar trend to F_1^k, it lends itself as it a good candidate for monitoring the Frisch-character of the solution. In fact, only $n_a + 1$ additional flops are required to compute F_3^k, since $\hat{r}_k$ is readily given by the conjugate gradient method. In addition, the fact that all measures tend to zero with increasing k indicates that, at least in the example considered, the solution of the RFSb scheme can approximately retain the Frisch-character of the solution, after the initialisation transients of the recursive scheme have decayed.

3.6 Critical appraisal and discussion

So far, recursive expressions for the computation of the Frisch-YW solution have been obtained. This section critically reviews the previous development and discusses potential shortcomings of the RFSa and RFSb algorithms while simultaneously evaluating the potential for improvements. Most of the ideas within this section are not considered further within this thesis, but indicate several directions for future work. Some aspects, concerning the reduction of the computation complexity, provide the motivation and lay the foundation for parts of the next Chapter.

3.6.1 Computation of $\sigma_{\tilde{y}}$

The output measurement noise variance is given as the smallest eigenvalue of the matrix $\hat{A}_k$, i.e.

$$\hat{A}_k x_k = \hat{\sigma}_{\tilde{y}}^k x_k \tag{3.61}$$

where x_k is the eigenvector corresponding to the eigenvalue $\hat{\sigma}_{\tilde{y}}^k$. The requirement to compute the eigenvalue in a recursive manner prompts the need for subspace tracking algorithms. These usually track the eigenvector x_k from which the eigenvalue is deduced via the Rayleigh quotient. In this chapter, a conjugate gradient method coupled with a line search has been utilised for this purpose. However, there exists a rich collection of subspace tracking algorithms within the literature which could replace the conjugate gradient method. Indeed, it is stated in (Feng & Owen 1996) that the conjugate gradient method is mainly the preferred choice for large scale problems, and that in the case of low dimensional problems (such as in the Frisch scheme where the eigenvalue of a $n_a + 1$ square matrix is to be determined) the inverse power iteration method is the recommended choice. However, the (iterative) conjugate gradient method exhibits good convergence properties, which justifies its choice for the RFS algorithms: The conjugate gradient method is globally convergent, when it is utilised for the minimisation of the Rayleigh quotient in an iterative manner (Yang 1993). It is also shown in

(Yang 1993), that the algorithm converges globally in a finite number of steps, provided the initial eigenvector estimate $\hat{x}_0$ contains an eigendirection of the true eigenvector. For the recursive CG-RQ algorithm within the RFS schemes, this means that repeated iterations at time k would not only improve the approximation of the smallest eigenvalue, but could eventually yield the exact solution (if desired). This would, in turn, imply that the Frisch-character is exactly satisfied.

Different approaches to estimate the output measurement noise variance are also feasible by exploiting the fact that the eigenvector x_k corresponding to $\hat{\sigma}_{\tilde{y}}^k$ is part of the parameter vector θ, as outlined in the following lemma.

Lemma 3.3. The eigenvector of $\Sigma_{\bar{\varphi}_0}$ corresponding to the smallest eigenvalue equal to zero is given by $\bar{\theta}$, whilst the eigenvector of

$$A \triangleq \Sigma_{\bar{\varphi}_y} - \Sigma_{\bar{\varphi}_y\varphi_u} \left[\Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b}\right]^{-1} \Sigma_{\varphi_u\bar{\varphi}_y} \tag{3.62}$$

corresponding to the minimal eigenvalue $\sigma_{\tilde{y}}$ is given by $\bar{a}$.

Proof. Since

$$\Sigma_{\bar{\varphi}_0}\bar{\theta} = 0 \tag{3.63}$$

holds, it is clear that the extended parameter vector $\bar{\theta}$ is the eigenvector of $\Sigma_{\bar{\varphi}_0}$ corresponding to the zero eigenvalue.

For the second part of the proof, (3.63) is re-expressed in block matrix form as

$$\begin{bmatrix} \Sigma_{\bar{\varphi}_y} - \sigma_{\tilde{y}} I_{n_a+1} & \Sigma_{\bar{\varphi}_y\varphi_u} \\ \Sigma_{\varphi_u\bar{\varphi}_y} & \Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b} \end{bmatrix} \begin{bmatrix} \bar{a} \\ b \end{bmatrix} = 0, \tag{3.64}$$

where the second block row yields

$$b = -\left[\Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b}\right]^{-1} \Sigma_{\varphi_u\bar{\varphi}_y}\bar{a}. \tag{3.65}$$

Substituting b in the upper part of (3.64) gives

$$\left(\Sigma_{\bar{\varphi}_y} - \Sigma_{\bar{\varphi}_y\varphi_u} \left[\Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b}\right]^{-1} \Sigma_{\varphi_u\bar{\varphi}_y}\right) \bar{a} = \sigma_{\tilde{y}}\bar{a}, \tag{3.66}$$

which concludes the proof. ∎

Taking the result of Lemma 3.3 into account, it becomes apparent that an estimate of $\bar{a}_k$ is computed twice within the RFS algorithms: firstly within the conjugate gradient algorithm (the eigenvector $\hat{x}_k$ which has to be scaled such that the first element becomes unity) and secondly within the bias compensating RLS scheme ($\hat{a}_k$ within $\hat{\theta}_k$). Consequently, there seems to be scope for improvement for a more efficient RFS implementation. This is investigated further in Chapter 4.

3.6.2 Minimisation of the YW cost function

One of the identified bottlenecks of the RFS algorithms in Section 3.4.2 is the computation of $\hat{\sigma}_{\tilde{u}}^k$. This requires $O(n_\theta^3)$ flops in its present form and this would appear to be not particularly attractive for a recursive scheme. By making use of stationary iterative least squares techniques (as implemented within the RBCLS; see also Section 2.3.1), it is, however, possible to reduce the overall computational complexity for both, the YW-GN and the YW-lin algorithms from cubic to quadratic order. This is also investigated further in Chapter 4.

3.6.3 Relationship to iterative bias eliminating schemes

Remark 3.6 has highlighted the close relationship between the iterative BCLS techniques and the novel RFS algorithms developed in this chapter. The underlying idea of the iterative BCLS algorithms is that of recursive bias compensation via RLS, when the noise variances are known. Iterative/recursive bias eliminating least squares (BELS) algorithms (Zheng & Feng 1989, Zheng 1998) also utilise the RBCLS concept. In addition they estimate $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$ in a recursive fashion. Consequently, the computation of θ within the BELS and RFS algorithms is identical and the only difference is the manner in which $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$ are computed. Moreover, it has recently been shown in (Hong & Söderström 2008) that the $\lambda_{\min}$-equation (3.1b) for the determination of $\sigma_{\tilde{y}}$ and the expression used within the BELS algorithms are both equivalent. In addition, the extended model within the BELS can be chosen in a manner, such that the resulting equations for $\sigma_{\tilde{u}}$ become identical to the YW equations. In this particular case, BELS and Frisch-YW use the same equations. This highlights the very close relationship of both approaches and shows, in fact, that the only difference between the Frisch-YW scheme and the BELS with an appropriately chosen extended model is only of an algorithmic nature. This implies that the asymptotic accuracy of these methods (in the case of convergence) is also identical. However, it has been shown in (Söderström, Hong & Zheng 2005) that the iterative BELS implementation might suffer from convergence problems in the cases of low signal-to-noise ratio. So far, the convergence properties of the RFS algorithms have not been investigated, but it would seem likely that they might suffer from the same problems.

3.6.4 Computation of θ

As outlined in Section 3.6.3, the RBCLS algorithm for the determination of $\hat{\theta}_k$, which is based on the principle of stationary iterative methods for LS, might suffer from convergence problems in the case of low signal-to-noise ratios. In order to investigate this issue (at least via simulation), the following example is considered.

Example 3.3. Consider a similar setup as in Section 3.3.6, i.e. the system is defined

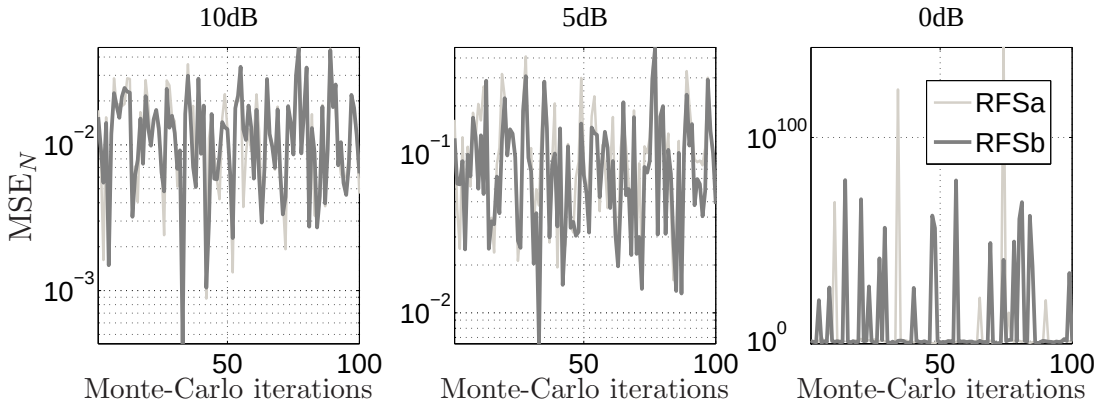


Figure 3.6: Mean square error of the RFSa and RFSb estimates for different signal-to-noise ratios.

once again by

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix}^T, \tag{3.67}$$

whilst $n_\zeta = n_a + n_b + 1$, $\lambda = 1$ and the input is a zero mean, white random process of unity variance. Three different signal-to-noise ratios (equal on input and output) are considered: 10dB, 5dB and 0dB. The system is simulated for $N = 500$ samples and the RFSa and RFSb algorithms are applied to estimate ϑ for 100 Monte-Carlo simulations. At each Monte-Carlo run, the mean squared error (MSE), defined by

$$\mathrm{MSE}_N \triangleq \frac{\left\| \vartheta - \hat{\vartheta}_N \right\|_2^2}{n_\theta + 2} \tag{3.68}$$

is computed and stored. The results are presented in Figure 3.6. It is observed that for the cases of 10dB and 5dB, the MSE lies within acceptable regions, although its average is larger in the latter case, which is expected due to the increased noise level. For the case of 0dB, however, several outliers are observed, indicating the divergence of the recursive scheme. Although a noise level of zero dB might be rather unrealistic in practice, the example shows that the RFS algorithms, in their present form, cannot guarantee convergence in general. ■

Example 3.3 shows that the RFS algorithms seem to suffer from convergence problems for low signal-to-noise ratios. Therefore, it appears reasonable, to consider alternative approaches for the computation of the bias compensated parameter vector. Indeed, if the RBCLS algorithm is replaced with a numerically sound algorithm (e.g. with the offline LS as discussed below), the modified RFS algorithms show no divergence when applied to Example 3.3. Some potential alternatives to compute $\hat{\theta}_k$, which are, however, not further investigated within this thesis, are listed as follows.

1. **(LS)** The LS method can be applied to solve the bias compensated normal equations in an offline manner at each recursion, yielding the minimum norm solution

$$\hat{\theta}_k = \left(\hat{\Sigma}_{\varphi}^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k)\right)^{\dagger} \hat{\xi}_{\varphi y}^k. \tag{3.69}$$

   Such an approach is feasible since the number equations is fixed (not growing with time). A potential drawback could be that of the increased computational costs since the computational complexity of the LS algorithm is generally of cubic order. This requires, however, some further investigations.

2. **(Matrix factorisation)** It is possible to represent the covariance matrices in factorised form using Cholesky decomposition or *UD*-factorisation. Whilst this is known to improve the conditioning of the problem, it also allows an easy introduction of regularisation (Ljung 1999, p. 383). Since the bias compensation can be regarded as a form of de-regularisation (constants are subtracted from the diagonal rather than added as in the case of regularisation), the use of matrix factorisations should provide the means for a more robust recursive bias compensation (rather than using the technique of stationary iterative LS as in the case of the RBCLS).

3. **(Eigendecomposition)** As outlined in Section 3.6.1, a is already estimated within the conjugate gradient subspace tracking algorithm as the eigenvector of $\hat{A}_k$ corresponding to the eigenvalue $\sigma_{\tilde{y}}$. From (3.64) it is observed, that b could be determined by solving the overdetermined system of equations

$$\begin{bmatrix} \Sigma_{\bar{\varphi}_y \varphi_u} \\ (\Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b}) \end{bmatrix} b = - \begin{bmatrix} \left(\Sigma_{\bar{\varphi}_y} - \sigma_{\tilde{y}} I_{n_a+1}\right) \\ \Sigma_{\varphi_u \bar{\varphi}_y} \end{bmatrix} \bar{a}. \tag{3.70}$$

   If LS is utilised to estimate b, the problem reduces to $O(n_b^3)$ (in contrast to $O(n_\theta^3)$ as in the direct LS approach of Point 1 above).

Whilst a further investigation of the convergence properties of the RFS algorithms is not considered within this thesis, the above points are interesting candidates for immediate further work.

3.6.5 Extension to other Frisch scheme forms

Recall from Section 2.4.3 that the YW cost function is only one possibility to single out (or choose) a model from the set of admissible Frisch scheme solutions. The YW criterion has been chosen within this chapter, since the required covariance matrix $\Sigma_{\zeta\bar{\varphi}}$ can be obtained recursively in a straightforward manner and without loss of information. In contrast, when use is made of the covariance-match (CM) criterion, the auto-covariance sequence of the residuals is required to be updated. Indeed, the residuals at time k

cannot be computed recursively, since they require the knowledge of the entire data sequence $\{u_i, y_i\}_{i=1}^{k}$. However, it is possible to approximate the residuals within the recursive scheme using the current data and the most recent estimate of θ, which allows a finite dimensional data vector to be stored (cf. Meyer et al. 2006, Linden et al. 2006). A similar philosophy is also used within the recursive prediction error methods for the determination of the gradient (see Ljung 1999, p. 371). Consequently the developments within this chapter require only minor modifications in order to handle the Frisch-CM case. The development of recursive Frisch-CM algorithms is, therefore, another topic of potential further work.

3.7 Concluding remarks

The Frisch scheme for dynamic system identification, which utilises the Yule-Walker (YW) model selection criterion, has been modified to recursively estimate the parameters and measurement noise variances of linear time-invariant single-input single-output errors-in-variables systems. Two recursive algorithms have been developed which are denoted RFSa and RFSb, respectively. The system parameter vector is obtained by making use of the recursive bias compensating least squares (RBCLS) principle, which removes the asymptotic bias of the recursive least squares (RLS) estimates at each time instance. The eigenvalue problem, which is required to be solved for the determination of the output measurement noise variance, is recursively computed by making use of a conjugate gradient method. For the computation of the input measurement noise variance, the YW cost function is required to be minimised at each recursion step. In order to achieve this, two approaches have been considered within this Chapter:

1. A Gauss-Newton algorithm, denoted YW-GN, which makes use of approximate first and second order derivatives.

2. A steepest gradient algorithm, denoted YW-lin, where the optimal step size is determined via a line search. When linearisations of the offline Frisch scheme equations are utilised to approximate the gradient, this approach is equivalent to minimising a modified cost function, where the Frisch equations are replaced by their linearisations around the most recent estimates.

These two techniques for the determination of the input measurement noise variance have led to the proposition of two recursive Frisch scheme (RFS) algorithms, which are denoted RFSa and RFSb, respectively. These algorithms have been compared with the offline Frisch scheme in simulation, illustrating that they are able to approximate the offline Frisch scheme estimates. A detailed overview of the utilised algorithms and sub-routines of this Chapter is given in Table 3.7.

The computational complexity of both recursive algorithms has been analysed in terms of floating point operations (flops). It is shown that the recursive algorithms are

Alg.	Name	Description
3.1	RAFS	Repeatedly applied (offline) Frisch scheme.
3.2	RBCLS	Recursive bias compensating least squares algorithm. Compensates for the RLS bias at each recursion.
3.3	RLS	Normalised gain RLS. Scales the covariance matrix, such that $P_k = [\hat{\Sigma}_{\varphi}^{k}]^{-1}$.
3.4	CG-RQ	Conjugate gradient algorithm which tracks the minimum of the Rayleigh quotient.
3.5	YW-GN	Gauss-Newton algorithm which tracks the minimum of the YW cost function.
3.6	YW-lin	Gives closed-form solution of minimum of modified YW cost function, which makes use of the linearised Frisch scheme equations.
3.7	RFSa	First recursive Frisch scheme algorithm. Uses the RBCLS, CG-RQ and the YW-GN.
3.8	RFSb	Second recursive Frisch scheme algorithm. Uses the RBCLS, CG-RQ and the YW-lin.

Table 3.7: Overview of developed algorithms for Chapter 3 (Abbreviation Alg. denotes Algorithm).

of cubic complexity with respect to the number of system parameters to be estimated, i.e. they require $O(n_\theta^3)$ flops. The corresponding bottlenecks of the RFS algorithms have been identified and pointed out. Whilst the order of complexity of the recursive algorithms is identical to that of the repeatedly applied offline Frisch scheme (RAFS), it has been shown in simulation, however, that the recursive algorithms reduce the absolute computation time per recursion significantly. Therefore, the recursive schemes, which have been developed within this chapter, would appear to be more suitable for a practical online implementation.

In addition to the computational complexity, the so-called Frisch-character, which is a unique feature of the offline Frisch scheme, has been analysed when use is made of the RFS algorithms. Since a subspace tracking algorithm is utilised to approximate the output measurement noise variance, the Frisch-character of the solution holds only approximately for the recursive algorithms. Different measures have been introduced to quantify the Frisch-character of the recursive solution, and its efficiency for online computation has been discussed. It has been shown that this characteristic may be reflected by the residual, which is computed by the conjugate gradient method implicitly. Hence, a computationally inexpensive measure for the Frisch-character is available online. A further numerical example has shown, at least in the specific case considered, that the recursive algorithms are able to retain the Frisch-character after the initialisation transients have decayed. Hence, the developed algorithms are able to approximate the offline solution reasonably well.

Finally, a critical appraisal and discussion has highlighted potential shortcomings of the recursive schemes. Directions for further refinement have been outlined concerning the reduction of computational effort. Moreover, a Monte-Carlo simulation has revealed that the recursive algorithms appear to suffer from convergence problems in the case of low signal-to-noise ratios. Alternatives have been discussed, in order to potentially

improve the convergence properties of the algorithms, which provide the bases for further developments. A thorough convergence analysis for the RFS algorithms also remains an interesting topic for future work. Finally, the very close relationship of the RFS algorithms with existing iterative bias-eliminating schemes has been discussed.

Chapter 4

Fast algorithms and coloured output noise

Contents

Nomenclature

a_{LS}	Least squares solution of a
b_{LS}	Least squares solution of b
$\hat{A}_k$	Schur complement
$B_k(\alpha_k)$	Schur complement
G_k	Auxiliary matrix

h_k	Auxiliary vector
H_k	Auxiliary matrix
$J(\sigma_{\tilde{u}})$	Jacobian (of residual $s_k(\theta)$ with respect to $\sigma_{\tilde{u}}$)
J^*	Auxiliary matrix
$L_\theta(\vartheta)$	Linearised θ-equation of Frisch scheme
$L_{\sigma_{\tilde{y}}}(\vartheta)$	Linearised $\lambda_{\min}$-equation of Frisch scheme
$L_\theta^{k'}$	Recursively computed derivative of $L_\theta(\vartheta)$
MSE_k	Mean square error
$N(A)$	Nullspace of matrix A
P_k	Scaled covariance matrix obtained from recursive least squares
$r_k(\theta)$	Residual of Yule-Walker cost function
$s(\theta)$	Auxiliary vector
$S(\theta)$	Auxiliary matrix
V_1^k	Cost function for the determination of $\sigma_{\tilde{u}}$ (coloured output noise case)
$V_1^{k'}$, $V_1^{k''}$	First and second order derivative of V_1^k
V_k^{lin}	YW cost function using linearised Frisch scheme equations
V_2^k	Cost function for the determination of α_k (coloured output noise case)
$V_2^{k'}$, $V_2^{k''}$	First and second order derivative of V_2^k
V_{LS}	Asymptotic least squares criterion
x_k	Eigenvector corresponding to $\hat{A}_k$
α_k	Scaling factor
δ_k	Extended instrument vector comprising delayed inputs
δ_k^*	Obtained by deleting last entry in δ_k
$\iota(\vartheta)$	Auxiliary term
$\bar{\iota}(\vartheta)$	Auxiliary term
$\kappa(\vartheta)$	Auxiliary term
$\rho_{\tilde{y}}$	Vector of auto-correlation terms of $\tilde{y}_k$
ς	Input measurement noise variance (obtained by $\lambda_{\min}$ equation)
$\hat{\theta}_{k-\frac{1}{2}}$	Intermediate estimate of θ
$\theta_k^{(\tilde{y})}$	Approximate derivative of $\hat{\theta}_k$ with respect to $\hat{\sigma}_{\tilde{y}}^k$
$\theta_k^{(\tilde{u})}$	Approximate derivative of $\hat{\theta}_k$ with respect to $\hat{\sigma}_{\tilde{u}}^k$
Θ	Augmented parameter vector for coloured output noise case
ζ_k	Instrument vector comprising delayed inputs

Preliminary reading: Sections 2.2, 2.3, 2.4.3, 2.4.4, 3.3, 3.4.

4.1 Introduction

Chapter 3 has developed algorithms for the online computation of the estimates obtained by the Frisch scheme, which utilises the Yule-Walker (YW) model selection criterion (Frisch-YW). Whilst the simulation results are promising, the computational cost of the recursive Frisch scheme (RFS) algorithms is of cubic order. Consequently, with respect to practical applicability, it is considered to be pertinent to investigate potential improvements to reduce the computational complexity. Section 3.6 has already identified the bottlenecks within the RFS algorithms and the first part of this chapter develops two novel algorithms with reduced computational costs. In order to gain computational speed, sacrifices are made by utilising additional approximations,

which will effect the estimates to be obtained. The fast RFS (FRFS) algorithms are compared via simulation with the RFSa and RFSb algorithms, which have been developed in Chapter 3. The very close relationship of the developed FRFS algorithms to the family of bias eliminating least squares (BELS) algorithms is also discussed. The development of the FRFS algorithms is based partly on the author's work published in (Linden, Vinsonneau & Burnham 2007*b*).

In the second part of this chapter the assumption of the output noise being white is relaxed and the development of recursive algorithms for coloured output noise is considered. First an algorithm is proposed, which uses Newton's method, based on the offline Frisch scheme for the coloured output noise case, which has been developed in (Söderström 2006). This initial algorithm leads to a rather computationally demanding scheme, due to the computation required to obtain the first and second order derivatives. A second approach takes into account some recently proposed simplifications of the offline scheme (Söderström 2008). In addition, it makes use of the particular bilinear parametrisation structure for the subproblem, which is concerned with estimating the input noise variance. From a computational perspective, this yields a more appealing algorithm. Both approaches are compared via a simulation example. Parts of this work have been published in (Linden & Burnham 2008*a*).

4.2 Fast recursive Frisch scheme algorithms

This section develops fast variants of the RFSa and RFSb algorithms which have been developed in Chapter 3 (cf. Section 3.3.5). The bottlenecks for these algorithms have been identified in Section 3.4 and are summarised as follows.

- The computation of $\sigma_{\tilde{u}}$ within the RFSa algorithm is achieved via the YW-GN algorithm. The computational complexity of this algorithm is of cubic order due to the computation of the derivatives of $\hat{\theta}_k$ with respect to $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, denoted $\theta_k^{(\tilde{u})}$ and $\theta_k^{(\tilde{y})}$, respectively, which requires the inversion of the matrix $\Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1}) = \left(\hat{\Sigma}_{\varphi}^{k} - \hat{\Sigma}_{\tilde{\varphi}}^{k-1}\right)^{-1}$ (cf. Steps 1-2 in Table 3.3 on page 75).
- The computation of $\sigma_{\tilde{u}}$ within the RFSb algorithm is achieved via the YW-lin algorithm. The computational complexity of this algorithm is of cubic order also due to the inversion of $\Sigma_{\varphi_0}^{-1}(\hat{\sigma}_{k-1})$ (cf. Step 1 in Table 3.4 on page 75).
- The computation of $\sigma_{\tilde{y}}$ within both RFS algorithms is achieved via the CG-RQ algorithm. The computation of the Schur complement $\hat{A}_k$ is of cubic complexity due to the inversion of a $n_b \times n_b$ matrix (cf. Step 1 in Table 3.5 on page 76).

Consequently, alternative approaches are suggested within this section, which avoid the matrix inversions, thereby reducing the computational complexity of the algorithms from cubic to quadratic order. In addition, the similarity of the resulting algorithms

with existing bias eliminating least squares (BELS) approaches is highlighted and a numerical simulation study compares the novel FRFS algorithms with the RFS algorithms, which have been developed in Chapter 3.

4.2.1 Fast YW-GN algorithm

In Section 3.4 the bottleneck within the Gauss-Newton method has been identified to be the computation of the derivatives

$$\theta_k^{(\tilde{u})} = \left(\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right)^{-1} \begin{bmatrix} 0 \\ \hat{b}_{k-1} \end{bmatrix}, \tag{4.1a}$$

$$\theta_k^{(\tilde{y})} = \left(\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right)^{-1} \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix}. \tag{4.1b}$$

Due to the matrix inversion involved, the computations are of cubic complexity, i.e. $O(n_\theta^3)$. However, it is straightforward to derive a less computationally demanding recursive expression of (4.1), by making use of the principle of stationary iterative least squares (see Section 2.3.1), where the matrix splitting is given naturally by $\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})$. Focusing on the computation of $\theta_k^{(\tilde{u})}$, (4.1a) may be re-expressed as

$$\theta_k^{(\tilde{u})} = \left[\hat{\Sigma}_\varphi^k\right]^{-1} \left(\begin{bmatrix} 0 \\ \hat{b}_{k-1} \end{bmatrix} + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\theta_k^{(\tilde{u})} \right) \tag{4.2}$$

which can be solved in a recursive way. Note that it is not necessary to compute the matrix inverse explicitly since it corresponds to P_k within the RLS algorithm (see (3.12)). Proceeding with $\theta_k^{(\tilde{y})}$ in a similar fashion, recursive formulations for the derivatives are then given by

$$\breve{\theta}_k^{(\tilde{u})} = P_k \left(\begin{bmatrix} 0 \\ \hat{b}_{k-1} \end{bmatrix} + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\breve{\theta}_{k-1}^{(\tilde{u})} \right), \tag{4.3a}$$

$$\breve{\theta}_k^{(\tilde{y})} = P_k \left(\begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\breve{\theta}_{k-1}^{(\tilde{y})} \right), \tag{4.3b}$$

where the notation $\breve{}$ is introduced, in order to distinguish between the recursively computed gradients and the non-recursive gradient computation given in (3.34a)-(3.34b). Note that only $O(n_\theta^2)$ flops are required for the computation of (4.3). The corresponding Gauss-Newton algorithm is denoted YW-GN-fast and is summarised as follows.

Algorithm 4.1 (YW-GN-fast).

$$\hat{\sigma}_{\tilde{u}}^k = \hat{\sigma}_{\tilde{u}}^{k-1} - \gamma_k \left[J_k^{(r)\,T} J_k^{(r)} \right]^{-1} J_k^{(r)\,T} r_k(\hat{\theta}_{k-1}) \tag{4.4a}$$

Step	Description	Procedure	Flops
1	Derivative	$\breve{\theta}_k^{(\tilde{u})} = P_k \left(\begin{bmatrix} 0 & \hat{b}_{k-1}^T \end{bmatrix}^T + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \breve{\theta}_{k-1}^{(\tilde{u})} \right)$	$O(n_\theta^2)$
2	Derivative	$\breve{\theta}_k^{(\tilde{y})} = P_k \left(\begin{bmatrix} \hat{a}_{k-1}^T & 0 \end{bmatrix}^T + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \breve{\theta}_{k-1}^{(\tilde{y})} \right)$	$O(n_\theta^2)$
3	Derivative	$\sigma_{\tilde{y},k}^{(\tilde{y})} = -(\hat{b}_{k-1}^T \hat{b}_{k-1})/(\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1})$	$O(n_b + n_a)$
4	Jacobian	$J_k^{(r)^T} = \hat{\Sigma}_{\zeta\varphi}^k \left(\breve{\theta}_k^{(\tilde{y})} \sigma_{\tilde{y},k}^{(\tilde{u})} + \breve{\theta}_k^{(\tilde{u})} \right)$	$O(n_\theta)$
5	Update $\hat{\sigma}_{\tilde{u}}$	$\hat{\sigma}_{\tilde{u}}^k = \hat{\sigma}_{\tilde{u}}^{k-1} + \gamma_k \left[J_k^{(r)^T} J_k^{(r)} \right]^{-1} J_k^{(r)^T} r_k(\hat{\theta}_{k-1})$	$O(n_\theta)$
		Overall complexity (dominant parts)	$O(n_\theta^2)$

Table 4.1: Computational complexity of YW-GN-fast algorithm.

$$J_k^{(r)^T} = \hat{\Sigma}_{\zeta\varphi}^k \left(\breve{\theta}_k^{(\tilde{y})} \sigma_{\tilde{y},k}^{(\tilde{u})} + \breve{\theta}_k^{(\tilde{u})} \right) \tag{4.4b}$$

$$\sigma_{\tilde{y},k}^{(\tilde{y})} = -(\hat{b}_{k-1}^T \hat{b}_{k-1})/(\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}) \tag{4.4c}$$

$$\breve{\theta}_k^{(\tilde{u})} = P_k \left(\begin{bmatrix} 0 \\ \hat{b}_{k-1} \end{bmatrix} + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \breve{\theta}_{k-1}^{(\tilde{u})} \right) \tag{4.4d}$$

$$\breve{\theta}_k^{(\tilde{y})} = P_k \left(\begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) \breve{\theta}_{k-1}^{(\tilde{y})} \right) \tag{4.4e}$$

$$r_k(\hat{\theta}_{k-1}) = \hat{\Sigma}_{\zeta\varphi}^k \hat{\theta}_{k-1} - \hat{\xi}_{\zeta y}^k \tag{4.4f}$$

For completeness, the computational complexity of Algorithm 4.1 is given in Table 4.1.

The next subsection considers a fast implementation of the YW-lin algorithm.

4.2.2 Fast YW-lin algorithm

Before developing the fast algorithm it is instructive to begin with a brief review of the YW-lin algorithm. Then, the two bottlenecks within this algorithm are pointed out and modifications are proposed to reduce the computational complexity.

Review of the YW-lin algorithm

Recall from Section 3.3.4 that within the YW-lin algorithm, $\sigma_{\tilde{u}}$ is obtained by minimising an approximation of the YW model selection cost function (3.44)

$$V_k^{\text{lin}} \triangleq \frac{1}{2} \left\| r_k \left(L_\theta(\hat{\vartheta}_{k-1}) \right) \right\|_2^2, \tag{4.5}$$

where the residual is given by

$$r_k\left(L_\theta(\hat{\vartheta}_{k-1})\right) = \hat{\Sigma}^k_{\zeta\varphi} L_\theta(\hat{\vartheta}_{k-1}) - \hat{\xi}^k_{\zeta y}. \tag{4.6}$$

The linearised bias compensating least squares equation is given by (3.29)

$$L_\theta(\hat{\vartheta}_{k-1}) = \hat{\theta}_{k-1} + \Sigma^{-1}_{\varphi_0}(\hat{\sigma}_{k-1})\iota(\hat{\vartheta}_{k-1}) + \Sigma^{-1}_{\varphi_0}(\hat{\sigma}_{k-1})\kappa(\hat{\vartheta}_{k-1})\hat{\sigma}^k_{\tilde{u}}, \tag{4.7}$$

with

$$\iota(\hat{\vartheta}_{k-1}) = \hat{\xi}^k_{\varphi y} - \hat{\Sigma}^k_\varphi \hat{\theta}_{k-1} + \left[\hat{\sigma}^{k-1}_{\tilde{y}} + \frac{\hat{b}^T_{k-1}\hat{b}_{k-1}}{\hat{\bar{a}}^T_{k-1}\hat{\bar{a}}_{k-1}}\hat{\sigma}^{k-1}_{\tilde{u}}\right]\begin{bmatrix}\hat{a}_{k-1}\\ 0\end{bmatrix}, \tag{4.8a}$$

$$\kappa(\hat{\vartheta}_{k-1}) = \begin{bmatrix} -\frac{\hat{b}^T_{k-1}\hat{b}_{k-1}}{\hat{\bar{a}}^T_{k-1}\hat{\bar{a}}_{k-1}}\hat{a}_{k-1} \\ \hat{b}_{k-1}\end{bmatrix}, \tag{4.8b}$$

$$\Sigma_{\varphi_0}(\hat{\sigma}_{k-1}) = \hat{\Sigma}^k_\varphi - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}). \tag{4.8c}$$

The update equation for $\sigma_{\tilde{u}}$ is then obtained by solving (3.45)

$$0 = \frac{dV^{\text{lin}}_k}{d\hat{\sigma}^k_{\tilde{u}}} = J^T(\hat{\sigma}^k_{\tilde{u}}) r_k\left(L_\theta(\hat{\vartheta}_{k-1})\right), \tag{4.9}$$

for $\sigma_{\tilde{u}}$, where the Jacobian is given by (3.48)

$$J(\hat{\sigma}^k_{\tilde{u}}) = \hat{\Sigma}^k_{\zeta\varphi}\frac{dL_\theta}{d\hat{\sigma}^k_{\tilde{u}}}. \tag{4.10}$$

First bottleneck

The first bottleneck of this YW-lin algorithm is due to the computation of the total derivative of L_θ, which has been given by (3.49)

$$\frac{dL_\theta}{d\hat{\sigma}^k_{\tilde{u}}} = \Sigma^{-1}_{\varphi_0}(\hat{\sigma}_{k-1})\kappa(\hat{\vartheta}_{k-1}), \tag{4.11}$$

where the matrix inverse $\Sigma^{-1}_{\varphi_0}$ is required to be computed at each recursion step. However, by making use of stationary iterative methods for solving LS problems (cf. Section 2.3.1), (4.11) can be re-expressed as

$$\hat{\Sigma}^k_\varphi \frac{dL_\theta}{d\hat{\sigma}^k_{\tilde{u}}} - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\frac{dL_\theta}{d\hat{\sigma}^k_{\tilde{u}}} = \kappa(\hat{\vartheta}_{k-1}), \tag{4.12}$$

where the matrix splitting is given naturally by (4.8c). An iterative/recursive way to compute $dL_\theta/d\hat{\sigma}^k_{\tilde{u}}$ could therefore be given by

$$L^{k'}_\theta \triangleq P_k\left[\kappa(\hat{\vartheta}_{k-1}) + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})L^{k-1'}_\theta\right], \tag{4.13}$$

where $L_\theta^{k'}$ denotes the recursively computed derivative and $P_k = [\hat{\Sigma}_\varphi^k]^{-1}$ is given by the matrix inversion lemma of the RLS algorithm (see Algorithm 3.3).

Second bottleneck

The second bottleneck within the YW-lin algorithm is due to the matrix inverse within the computation of (4.7), therefore, a recursive expression for $L_\theta(\hat{\vartheta}_{k-1})$ is required. Firstly, introduce the notation $L_\theta(\hat{\vartheta}_{k-1}) \triangleq L_\theta^k$, where the index k is chosen to reflect the fact that L_θ^k corresponds to the linearisation at time instance k (although it depends on the estimate $\hat{\vartheta}_{k-1}$ with time index $k-1$). Secondly, assume that all past $\hat{\theta}_k$ have been computed using the expression (4.7), which means that $\hat{\theta}_k$ can be replaced with L_θ^k in (4.7). Thirdly, from (4.8a) and (4.8b) it holds

$$\begin{aligned}
\iota(\hat{\vartheta}_{k-1}) + \kappa(\hat{\vartheta}_{k-1})\hat{\sigma}_{\tilde{u}}^k &= \hat{\xi}_{\varphi y}^k - \hat{\Sigma}_\varphi^k \hat{\theta}_{k-1} + \left[\hat{\sigma}_{\tilde{y}}^{k-1} + \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{\sigma}_{\tilde{u}}^{k-1}\right] \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} \\
&\quad + \begin{bmatrix} -\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{a}_{k-1} \\ \hat{b}_{k-1} \end{bmatrix} \hat{\sigma}_{\tilde{u}}^k \\
&= \hat{\xi}_{\varphi y}^k - \hat{\Sigma}_\varphi^k \hat{\theta}_{k-1} + \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} \hat{\sigma}_{\tilde{y}}^{k-1} + \begin{bmatrix} \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{a}_{k-1} \\ 0 \end{bmatrix} \hat{\sigma}_{\tilde{u}}^{k-1} \\
&\quad + \begin{bmatrix} -\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{a}_{k-1} \\ 0 \end{bmatrix} \hat{\sigma}_{\tilde{u}}^k + \begin{bmatrix} 0 \\ \hat{b}_{k-1} \end{bmatrix} \hat{\sigma}_{\tilde{u}}^k,
\end{aligned} \tag{4.14}$$

and by assuming that $\hat{\sigma}_{\tilde{u}}^k \approx \hat{\sigma}_{\tilde{u}}^{k-1}$, $\hat{\sigma}_{\tilde{y}}^k \approx \hat{\sigma}_{\tilde{y}}^{k-1}$ and using $\hat{\theta}_{k-1} = L_\theta^{k-1}$, one obtains

$$\iota(\hat{\vartheta}_{k-1}) + \kappa(\hat{\vartheta}_{k-1})\hat{\sigma}_{\tilde{u}}^k \approx \hat{\xi}_{\varphi y}^k - \hat{\Sigma}_\varphi^k L_\theta^{k-1} + \begin{bmatrix} \hat{\sigma}_{\tilde{y}}^k I_{n_a} & 0 \\ 0 & \hat{\sigma}_{\tilde{u}}^k I_{n_b} \end{bmatrix} L_\theta^{k-1}. \tag{4.15}$$

Finally, by substituting (4.8c) and (4.15) into (4.7), it holds

$$\left[\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right] L_\theta^k = \left[\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right] L_\theta^{k-1} + \hat{\xi}_{\varphi y}^k - \hat{\Sigma}_\varphi^k L_\theta^{k-1} + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k) L_\theta^{k-1}, \tag{4.16}$$

which simplifies to

$$\left[\hat{\Sigma}_\varphi^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1})\right] L_\theta^k = -\Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) L_\theta^{k-1} + \hat{\xi}_{\varphi y}^k + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k) L_\theta^{k-1}. \tag{4.17}$$

Thus, a recursive computation of the linearised θ-equation is given by

$$\begin{aligned}
&\hat{\Sigma}_\varphi^k L_\theta^k \approx \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) L_\theta^{k-1} - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) L_\theta^{k-1} + \hat{\xi}_{\varphi y}^k + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k) L_\theta^{k-1} \\
\Leftrightarrow \quad &L_\theta^k \approx [\hat{\Sigma}_\varphi^k]^{-1} \hat{\xi}_{\varphi y}^k + [\hat{\Sigma}_\varphi^k]^{-1} \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k) L_\theta^{k-1},
\end{aligned} \tag{4.18}$$

which interestingly is, indeed, the recursive bias compensating least squares (RBCLS) algorithm given in Section 3.3.2 (i.e. simply replace $\hat{\theta}_k$ with L_θ^k in (3.10)). Since the recursive computation of L_θ^k is identical to the RBCLS computation of $\hat{\theta}_k$, the latter, more familiar, notation can be utilised. Substituting the linearised $\lambda_{\min}$-equation (3.27b), with the linearisation carried out around $\hat{\vartheta}_{k-1}$, the RBCLS equation becomes

$$\begin{aligned} & \hat{\theta}_k = \hat{\theta}_k^{\mathrm{LS}} + P_k \begin{bmatrix} \hat{\sigma}_{\tilde{y}}^k \hat{a}_{k-1} \\ \hat{\sigma}_{\tilde{u}}^k b_{k-1} \end{bmatrix} \\ \Leftrightarrow \quad & \hat{\theta}_k = \hat{\theta}_k^{\mathrm{LS}} + P_k \begin{bmatrix} 0 \\ \hat{b}_{k-1} \end{bmatrix} \hat{\sigma}_{\tilde{u}}^k + P_k \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} \left[\hat{\sigma}_{\tilde{y}}^{k-1} + \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{\sigma}_{\tilde{u}}^{k-1} - \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{\sigma}_{\tilde{u}}^k \right], \end{aligned} \tag{4.19}$$

which simplifies to

$$\hat{\theta}_k = P_k \bar{\iota}(\hat{\vartheta}_{k-1}) + P_k \kappa(\hat{\vartheta}_{k-1}) \hat{\sigma}_{\tilde{u}}^k, \tag{4.20}$$

where $\kappa(\hat{\vartheta}_{k-1})$ is defined by (4.8b) and

$$\bar{\iota}(\hat{\vartheta}_{k-1}) \triangleq \hat{\xi}_{\varphi y}^k + \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} \left[\hat{\sigma}_{\tilde{y}}^{k-1} + \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \hat{\sigma}_{\tilde{u}}^{k-1} \right]. \tag{4.21}$$

Fast update

Using the previous results, a fast implementation of the YW-lin algorithm can be realised. With the Jacobian at time instance k being given by (cf. (3.48))

$$J_k \triangleq \hat{\Sigma}_{\zeta\varphi}^k L_\theta^{k'}, \tag{4.22}$$

it is therefore possible to solve (4.9) as

$$\begin{aligned} 0 &= J_k^T \left[\hat{\Sigma}_{\zeta\varphi}^k \hat{\theta}_k - \hat{\xi}_{\zeta y}^k \right] \\ \Leftrightarrow \quad J_k^T \hat{\Sigma}_{\zeta\varphi}^k \left[P_k \bar{\iota}(\hat{\vartheta}_{k-1}) + P_k \kappa(\hat{\vartheta}_{k-1}) \hat{\sigma}_{\tilde{u}}^k \right] &= J_k^T \hat{\xi}_{\zeta y}^k \\ \Leftrightarrow \quad J_k^T \hat{\Sigma}_{\zeta\varphi}^k P_k \kappa(\hat{\vartheta}_{k-1}) \hat{\sigma}_{\tilde{u}}^k &= J_k^T \left[\hat{\xi}_{\zeta y}^k - \hat{\Sigma}_{\zeta\varphi}^k P_k \bar{\iota}(\hat{\vartheta}_{k-1}) \right], \end{aligned} \tag{4.23}$$

and the fast update for $\hat{\sigma}_{\tilde{u}}^k$ is finally given by

$$\hat{\sigma}_{\tilde{u}}^k = \frac{J_k^T \left[\hat{\xi}_{\zeta y}^k - \hat{\Sigma}_{\zeta\varphi}^k P_k \bar{\iota}(\hat{\vartheta}_{k-1}) \right]}{J_k^T \hat{\Sigma}_{\zeta\varphi}^k P_k \kappa(\hat{\vartheta}_{k-1})}. \tag{4.24}$$

The new algorithm, denoted YW-lin-fast, can be summarised as follows.

Step	Description	Procedure	Flops
1	Update $\hat{\sigma}_{\tilde{u}}^k$	$\hat{\sigma}_{\tilde{u}}^k = J_k^T \left[\hat{\xi}_{\zeta y}^k - \hat{\Sigma}_{\zeta\varphi}^k P_k \bar{\iota}(\hat{\vartheta}_{k-1})\right] / J_k^T \hat{\Sigma}_{\zeta\varphi}^k P_k \kappa(\hat{\vartheta}_{k-1})$	$O(n_\theta^2)$
2		$\kappa(\hat{\vartheta}_{k-1}) = \begin{bmatrix} -\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} & \hat{a}_{k-1} & \hat{b}_{k-1}^T \end{bmatrix}^T$	$O(n_{\bar{a}})$
3		$\bar{\iota}(\hat{\vartheta}_{k-1}) = \hat{\xi}_{\varphi y}^k + \left[\hat{\sigma}_{\tilde{y}}^{k-1} + \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} \hat{\sigma}_{\tilde{u}}^{k-1}\right] [\hat{a}_{k-1}^T \; 0]^T$	$O(n_\theta)$
4	Jacobian	$J_k = \hat{\Sigma}_{\zeta\varphi}^k L_\theta^{k'}$	$O(n_\theta^2)$
5	Derivative	$L_\theta^{k'} = P_k \left[\kappa(\hat{\vartheta}_{k-1}) + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) L_\theta^{k-1'}\right]$	$O(n_\theta^2)$
		Overall complexity (dominant parts)	$O(n_\theta^2)$

Table 4.2: Computational complexity of the YW-lin-fast algorithm.

Algorithm 4.2 (YW-lin-fast).

$$\hat{\sigma}_{\tilde{u}}^k = \frac{J_k^T \left[\hat{\xi}_{\zeta y}^k - \hat{\Sigma}_{\zeta\varphi}^k P_k \bar{\iota}(\hat{\vartheta}_{k-1})\right]}{J_k^T \hat{\Sigma}_{\zeta\varphi}^k P_k \kappa(\hat{\vartheta}_{k-1})} \tag{4.25a}$$

$$\kappa(\hat{\vartheta}_{k-1}) = \begin{bmatrix} -\frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} \hat{a}_{k-1} \\ \hat{b}_{k-1} \end{bmatrix} \tag{4.25b}$$

$$\bar{\iota}(\hat{\vartheta}_{k-1}) \triangleq \hat{\xi}_{\varphi y}^k + \left[\hat{\sigma}_{\tilde{y}}^{k-1} + \frac{\hat{b}_{k-1}^T \hat{b}_{k-1}}{\hat{a}_{k-1}^T \hat{a}_{k-1}} \hat{\sigma}_{\tilde{u}}^{k-1}\right] \begin{bmatrix} \hat{a}_{k-1} \\ 0 \end{bmatrix} \tag{4.25c}$$

$$J_k = \hat{\Sigma}_{\zeta\varphi}^k L_\theta^{k'} \tag{4.25d}$$

$$L_\theta^{k'} = P_k \left[\kappa(\hat{\vartheta}_{k-1}) + \Sigma_{\tilde{\varphi}}(\hat{\sigma}_{k-1}) L_\theta^{k-1'}\right] \tag{4.25e}$$

The number of flops for Algorithm 4.2 are listed in Table 4.2, where a reduction from cubic to quadratic complexity is observed (cf. Table 3.4 on page 75). Consequently, the computational complexity for determining $\sigma_{\tilde{u}}^k$ has been reduced by one order of magnitude.

The next subsection addresses the bottleneck within the conjugate gradient subspace tracking algorithm.

4.2.3 Alternative computations for $\hat{\sigma}_{\tilde{y}}$

Rayleigh quotient based approach

The RFS algorithms utilise a gradient-based subspace tracking algorithm, namely the conjugate gradient method described in Section 3.3.3, in order to recursively estimate the eigenvector corresponding to the smallest eigenvalue of $\hat{A}_k$. Once the eigenvector x_k is determined, the corresponding eigenvalue can be computed using the Rayleigh

quotient (cf. Golub & Van Loan 1996)

$$\hat{\sigma}_{\tilde{y}}^{k} = \frac{x_k^T \hat{A}_k x_k}{x_k^T x_k}, \tag{4.26}$$

as in (3.15c). A computationally less demanding approach for a recursive determination of $\hat{\sigma}_y^k$, which avoids the use of the subspace tracking algorithm, seems possible by acknowledging the fact that in the asymptotic case, the eigenvector of A corresponding to the smallest eigenvalue is already contained in the parameter vector, as pointed out in Lemma 3.3 (cf. page 83). Consequently, by assuming that $\hat{\bar{a}}_{k-1}$ computed by the recursive bias compensating least squares (RBCLS) is an approximation of the eigenvector of $\hat{A}_{k-1}$ corresponding to the smallest eigenvalue $\hat{\sigma}_{\tilde{y}}^{k-1}$, and by further assuming that $\hat{\bar{a}}_{k-1}$ is sufficiently 'close' to $\hat{\bar{a}}_k$, the minimum eigenvalue of $\hat{A}_k$ can be approximated using the Rayleigh quotient, which gives rise to the recursive expression

$$\hat{\sigma}_{\tilde{y}}^{k} = \frac{\hat{\bar{a}}_{k-1}^T \hat{A}_k \hat{\bar{a}}_{k-1}}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}}, \tag{4.27}$$

which completely avoids the need for tracking the eigenvector via the conjugate gradient algorithm.

Approximation of the Schur complement

Whilst the use of equation (4.27) does indeed reduce the required number of flops, the order of computational complexity does not change. This is due to the cubic complexity for the computation of the Schur complement $\hat{A}_k$ itself, which is the bottleneck within the conjugate gradient algorithm (see Step 1 in Table 3.5 on page 76). In order to reduce the order of complexity which is required for the computation of $\hat{A}_k$, a further approximation might be utilised. Recall, that in the asymptotic case, the Schur complement is given by (3.62)

$$A = \Sigma_{\bar{\varphi}_y} - \Sigma_{\bar{\varphi}_y \varphi_u} \left[\Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b}\right]^{-1} \Sigma_{\varphi_u \bar{\varphi}_y}, \tag{4.28}$$

whilst b can be expressed as (3.65)

$$b = -\left[\Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b}\right]^{-1} \Sigma_{\varphi_u \bar{\varphi}_y} \bar{a}. \tag{4.29}$$

Post-multiplying (4.28) with $\bar{a}$ and substituting (4.29) gives

$$A\bar{a} = \Sigma_{\bar{\varphi}_y} \bar{a} + \Sigma_{\bar{\varphi}_y \varphi_u} b, \tag{4.30}$$

which allows the output measurement noise variance, by using the Rayleigh quotient, to be expressed as

$$\sigma_{\tilde{y}} = \frac{\bar{a}^T}{\bar{a}^T\bar{a}} \left(\Sigma_{\bar{\varphi}_y} \bar{a} + \Sigma_{\bar{\varphi}_y \varphi_u} b \right). \tag{4.31}$$

This gives rise to an approximate recursive update for $\sigma_{\tilde{y}}$ given by

$$\hat{\sigma}_{\tilde{y}}^k = \frac{\hat{\bar{a}}_{k-1}^T}{\hat{\bar{a}}_{k-1}^T \hat{\bar{a}}_{k-1}} \left(\hat{\Sigma}_{\bar{\varphi}_y}^k \hat{\bar{a}}_{k-1} + \hat{\Sigma}_{\bar{\varphi}_y \varphi_u}^k \hat{b}_{k-1} \right), \tag{4.32}$$

which avoids the explicit computation of the Schur complement and only requires $2n_{\bar{a}}^2 + 2n_{\bar{a}}n_b + 6n_{\bar{a}}$ flops. Expression (4.32) does not, however, exploit the knowledge of the previously determined $\hat{\sigma}_{\tilde{u}}^k$, since it only depends on $\hat{\theta}_{k-1}$. Is seems reasonable, therefore, to perform an intermediate step, in order to make use of the most recent input noise variance estimate. This is achieved by computing an intermediate parameter estimate, denoted $\hat{\theta}_{k-\frac{1}{2}}$, which makes use of the most recent estimate of $\sigma_{\tilde{u}}$. The resulting algorithm, which is denoted ARQ (approximate Rayleigh quotient), can be summarised as follows.

Algorithm 4.3 (ARQ).

$$\hat{\theta}_{k-\frac{1}{2}} = \hat{\theta}_k^{\mathrm{LS}} + P_k \begin{bmatrix} \hat{\sigma}_{\tilde{y}}^{k-1} I_{n_a} & 0 \\ 0 & \hat{\sigma}_{\tilde{u}}^k I_{n_b} \end{bmatrix} \hat{\theta}_{k-1} \tag{4.33a}$$

$$\hat{\sigma}_{\tilde{y}}^k = \frac{\hat{\bar{a}}_{k-\frac{1}{2}}^T}{\hat{\bar{a}}_{k-\frac{1}{2}}^T \hat{\bar{a}}_{k-\frac{1}{2}}} \left(\hat{\Sigma}_{\bar{\varphi}_y}^k \hat{\bar{a}}_{k-\frac{1}{2}} + \hat{\Sigma}_{\bar{\varphi}_y \varphi_u}^k \hat{b}_{k-\frac{1}{2}} \right) \tag{4.33b}$$

The ARQ constitutes an update equation for $\hat{\sigma}_{\tilde{y}}^k$, where $\hat{\bar{\theta}}_{k-\frac{1}{2}}^T = [\hat{\bar{a}}_{k-\frac{1}{2}}^T \ \hat{b}_{k-\frac{1}{2}}^T]$. The overall fast RFS algorithms, which make use of the ARQ as well as the fast algorithms for the computation of $\hat{\sigma}_{\tilde{u}}^k$, are summarised in the following subsection.

4.2.4 Fast recursive Frisch scheme algorithms

Based on the above approximations and modifications, two fast RFS algorithms can be proposed. They are, analogously to the RFSa and RFSb, denoted FRFSa and FRFSb, respectively. The FRFSa makes use of the YW-GN-fast, whilst the FRFSb utilises the YW-lin-fast. Both algorithms use the ARQ algorithm for the computation of the output measurement noise variance. In contrast to the RFS algorithms developed in Chapter 3, the overall complexity of the FRFS algorithms is reduced to $O(n_\theta^2)$. For completeness, both algorithms are summarised as follows.

Algorithm 4.4 (FRFSa).

1) Initialisation

2) For $k = n_\zeta + n_b + 1, ...$

a) Update γ_k via (3.8) and $\hat{\Sigma}^k_{\bar{\varphi}}$ and $\hat{\Sigma}^k_{\zeta\bar{\varphi}}$ via (3.7)

b) Compute P_k and $\hat{\theta}^{\mathrm{LS}}_k$ using Algorithm 3.3

c) Update $\hat{\sigma}^k_{\tilde{u}}$ via Algorithm 4.1

d) Update $\hat{\sigma}^k_{\tilde{y}}$ using Algorithm 4.3

e) Compute $\hat{\theta}_k$ via Algorithm 3.2

Algorithm 4.5 (FRFSb).

1) Initialisation

2) For $k = n_\zeta + n_b + 1, ...$

a) Update γ_k via (3.8) and $\hat{\Sigma}^k_{\bar{\varphi}}$ and $\hat{\Sigma}^k_{\zeta\bar{\varphi}}$ via (3.7)

b) Compute P_k and $\hat{\theta}^{\mathrm{LS}}_k$ using Algorithm 3.3

c) Update $\hat{\sigma}^k_{\tilde{u}}$ via Algorithm 4.2

d) Update $\hat{\sigma}^k_{\tilde{y}}$ using Algorithm 4.3

e) Compute $\hat{\theta}_k$ via Algorithm 3.2

A detailed description providing the computational complexities of the FRFS algorithms is provided in Table 4.3.

Remark 4.1 (Linearised Frisch equations). A similar algorithm with an identical order of complexity could be defined by making use of the linearised $\lambda_{\min}$ equation (3.27b) rather than (4.32). In general it might seem appealing to make use of both linearised Frisch equations in order to obtain a 'cheaply to compute' recursive Frisch scheme algorithm. Whilst it has been shown in Section 4.2.2 that a recursive version of the linearised θ equation leads in fact to the RBCLS scheme, combining the linearised $\lambda_{\min}$ equation with the RBCLS and the fast YW-GN or YW-lin algorithm does not appear to perform well in simulation. One possible explanation could be that the linear approximation of $\sigma_{\tilde{y}}$ is not accurate enough to yield an overall satisfactory performance.

Step	Description	Procedure	Flops
1	Choose n_ζ, λ_k and j	$n_\zeta = n_a + n_b + 1$, $0 < \lambda_k < 1$, $j = n_\zeta + n_b$	
2	RLS initialisations	$\hat{\theta}^{\mathrm{LS}}_{n_a} = 0$, $P_{n_a} = 0.1I$	
3	Recursion	for $k = n_a + 1, ..., j$	
3.1	Data weighting	$\gamma_k = 1/k$	
3.2	RLS	Compute L_k, $\hat{\theta}^{\mathrm{LS}}_k$ and P_k as in Table 3.2	
4	General initialisations	$\hat{\Sigma}^j_{\zeta\bar{\varphi}} = 0$, $\hat{\Sigma}^j_{\bar{\varphi}} = \frac{1}{n_\zeta - 1}\sum_{i=n_b+1}^{j} \bar{\varphi}_i \bar{\varphi}_i^T$, $\gamma_j = 1/(n_\zeta - 1)$, $\hat{\sigma}^j_{\tilde{u}} = 0$, and $L^{j'}_\theta = 0$	
5	Recursion	for $k = j + 1, ...$	
5.1	Update γ	$\gamma_k = \frac{\gamma_{k-1}}{\lambda_k + \gamma_{k-1}}$	2
5.2	Update $\hat{\Sigma}_{\bar{\varphi}}$	$\hat{\Sigma}^k_{\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\bar{\varphi}} + \gamma_k \left(\bar{\varphi}_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\bar{\varphi}} \right)$	$O(n_\theta^2)$
5.3	Update $\hat{\Sigma}_{\zeta\bar{\varphi}}$	$\hat{\Sigma}^k_{\zeta\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}} + \gamma_k \left(\zeta_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}} \right)$	$O(n_\theta^2)$
5.4	RLS estimate	Compute L_k, $\hat{\theta}^{\mathrm{LS}}_k$ and P_k as in Table 3.2	$O(n_\theta^2)$
5.5	Fast update of $\hat{\sigma}^k_{\tilde{u}}$		
	either FRFSa:	Compute $\hat{\sigma}^k_{\tilde{u}}$ as in Table 4.1	$O(n_\theta^2)$
	or FRFSb:	Compute $\hat{\sigma}^k_{\tilde{u}}$ as in Table 4.2	$O(n_\theta^2)$
5.6	Projection	Project $0 \leq \hat{\sigma}^k_{\tilde{u}} \leq \sigma^{\max}_{\tilde{u}}$	
5.7	Intermediate update of $\hat{\theta}_k$	$\hat{\theta}_{k-\frac{1}{2}} = \hat{\theta}^{\mathrm{LS}}_k + P_k \mathrm{diag}(\hat{\sigma}^{k-1}_{\tilde{y}}, ..., \hat{\sigma}^k_{\tilde{u}}, ...)\hat{\theta}_{k-1}$	$O(n_\theta)$
5.8	Fast $\hat{\sigma}_y$ computation	$\hat{\sigma}^k_{\tilde{y}} = \hat{\bar{a}}^T_{k-\frac{1}{2}} / (\hat{\bar{a}}^T_{k-\frac{1}{2}} \hat{\bar{a}}_{k-\frac{1}{2}}) \left(\hat{\Sigma}^k_{\bar{\varphi}_y} \hat{\bar{a}}_{k-\frac{1}{2}} + \hat{\Sigma}^k_{\bar{\varphi}_y \varphi_u} \hat{b}_{k-\frac{1}{2}} \right)$	$O(n_{\bar{a}}^2)$
5.9	Projection	Project $0 \leq \hat{\sigma}^k_{\tilde{y}} \leq \sigma^{\max}_{\tilde{y}}$	
5.10	Bias compensation	$\hat{\theta}_k = \hat{\theta}^{\mathrm{LS}}_k + P_k \hat{\Sigma}^k_{\bar{\varphi}} \hat{\theta}_{k-1}$	$O(n_\theta^2)$
		Overall complexity (dominant parts)	$O(n_\theta^2)$

Table 4.3: Computational complexity of the FRFSa/FRFSb algorithm.

4.2.5 Relation between FRFS and BELS

Note that the expression (4.32) for the determination of the output measurement noise variance, which has been derived from the Rayleigh quotient (4.26), could also have been obtained from the first n_a equations of

$$\Sigma_{\varphi 0} \bar{\theta} = 0. \tag{4.34}$$

By taking the upper block row of (3.64), one obtains

$$\sigma_{\tilde{y}} \bar{a} = \Sigma_{\bar{\varphi}_y} \bar{a} + \Sigma_{\bar{\varphi}_y \varphi_u} b, \tag{4.35}$$

and pre-multiplying with $\bar{a}^T$ yields

$$\sigma_{\tilde{y}} = \frac{\bar{a}^T}{\bar{a}^T \bar{a}} \left(\Sigma_{\bar{\varphi}_y} \bar{a} + \Sigma_{\bar{\varphi}_y \varphi_u} b \right), \tag{4.36}$$

which is essentially (4.32). As already highlighted in Section 3.6.3, the Frisch scheme, the bias eliminating schemes and the extended bias compensating least squares (cf.

Chapter 5) are very closely related, since they all, with suitable choice of instruments and extended model, use the same equations as the Frisch-YW method. The developed FRFS algorithms resemble even more the bias eliminating least squares (BELS) approaches, since the latter also determine $\sigma_{\tilde{y}}$ in an iterative/recursive way, as the FRFS algorithms using (4.36). Although the FRFS algorithms within this section have been derived from a different perspective, using the Rayleigh quotient to solve the eigenvalue problem associated with the Frisch scheme, the very close relationship to the BELS algorithms is apparent.

4.2.6 Numerical examples

It is of interest to compare the FRFS approaches with the more computational demanding RFS algorithms for a particular example. Also, the effect of the departure (or otherwise) of the Frisch-character by introducing the various approximations for the development of the FRFS schemes is investigated in this subsection. Finally it appears natural to study the reduction of computational costs.

Example 4.1 (Estimation of $\sigma_{\tilde{u}}$). Consider a similar setup as in Section 3.3.6, where a LTI SISO system with $n_a = n_b = 2$ and given by

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix}^T \tag{4.37a}$$

$$\sigma = \begin{bmatrix} 2.1 & 0.1 \end{bmatrix}^T \tag{4.37b}$$

is simulated for 1000 samples using a zero mean, white and Gaussian distributed input signal of unity variance. The RFS and FRFS algorithms are applied to estimate ϑ using $n_\zeta = n_a + n_b + 1$, whilst $\lambda = 1$ is chosen (i.e. no forgetting). The maximal admissible values for the input and output measurement noise variances are chosen to be $\sigma_{\tilde{u}}^{\max} = 2\sigma_{\tilde{u}} = 0.2$ and $\sigma_{\tilde{y}}^{\max} = 2\sigma_{\tilde{y}} = 4.2$. The estimates of $\sigma_{\tilde{u}}$ are compared in Figure 4.1. The upper plot shows the results of the Gauss-Newton approaches RFSa and FRFSa in comparison to the RAFS. It is observed that the projection facility of the recursive algorithms is active during the first 200 recursions. After this period, the RFSa seems only to be able to slowly follow the changes of the RAFS, as already pointed out in Section 3.3.6. The FRFSa, which uses the recursively updated derivatives, seems to be slightly more sluggish. This becomes even more evident, if the $\sigma_{\tilde{u}}^{\max}$ is set to larger upper bound and the experiment is repeated. The lower plot of Figure 4.1 shows the estimate of $\sigma_{\tilde{u}}$ obtained from the RFSb and FRFSb algorithms. Here it is observed that the projection facility seems to be more often active for the fast algorithm (see around $k = 420$). After approximately 500 recursions, however, the FRFSb estimate is barely distinguishable from the RFSb and RAFS estimate, although the FRFSb solution seems to be slightly more erratic. ■

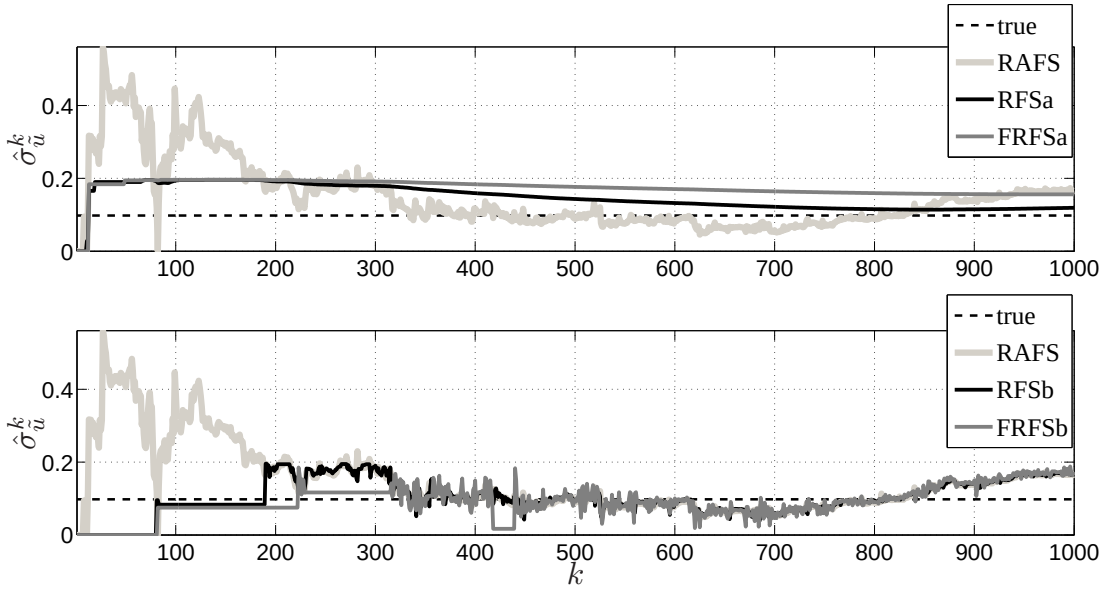


Figure 4.1: Estimates of $\sigma_{\tilde{u}}$ for using the RAFS, RFSa, RFSb, FRFSa and FRFSb.

The following example investigates, how well the ARQ algorithm can approximate the Frisch-character.

Example 4.2 (Estimation of $\sigma_{\tilde{y}}$). In order to investigate the Frisch-character of the solution, the smallest eigenvalue of $\hat{\Sigma}^k_{\varphi_0} = \hat{\Sigma}^k_{\varphi} - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k)$ (cf. (3.58)) evolving over time is observed[1]. Therefore, a similar setup as in Example 4.1 is considered. The RFSb and the corresponding FRFSb are applied to estimate ϑ. However, the value of $\hat{\sigma}^k_{\tilde{u}}$ is identical in both cases (obtained by the YW-lin algorithm), which allows the direct comparison of $\hat{\sigma}^k_{\tilde{y}}$ obtained from the CG-RQ (cf. Algorithm 3.4 on page 61) and that obtained from the ARQ algorithm. The smallest eigenvalue of $\hat{\Sigma}^k_{\varphi_0}$ using both algorithms is shown in Figure 4.2. It is observed, that both estimates of $\sigma_{\tilde{y}}$ are virtually identical after $k = 300$ recursions. However, it becomes apparent that there is a price to pay for the reduction of computational complexity for the computation of $\sigma_{\tilde{y}}$: The approximation of the Frisch-character is worse in the case of the FRFSb algorithm, which uses the fast ARQ update for the computation of the output measurement noise variance. ■

The final example investigates the computational savings when use is made of the fast algorithms.

Example 4.3 (Computation time). Naturally, it is of major interest to compare the computation time per recursion of the FRFS algorithms with those of the RFS schemes. Therefore, the experiment in Section 3.4.3 is repeated (cf. Figure 3.3 on page 78) for the FRFS algorithms. The results are presented in Figure 4.3, which clearly shows the reduction of computational complexity for the FRFS approaches. For the fast algorithms,

[1] Recall that the smallest eigenvalue is zero in the offline Frisch scheme case.

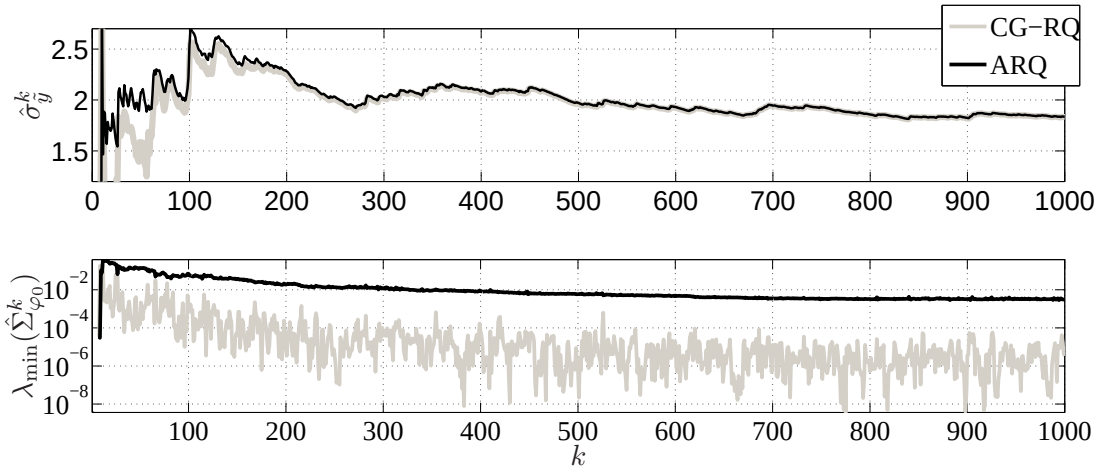


Figure 4.2: Estimates of the CG-RQ and ARQ algorithms and corresponding Frisch-character.

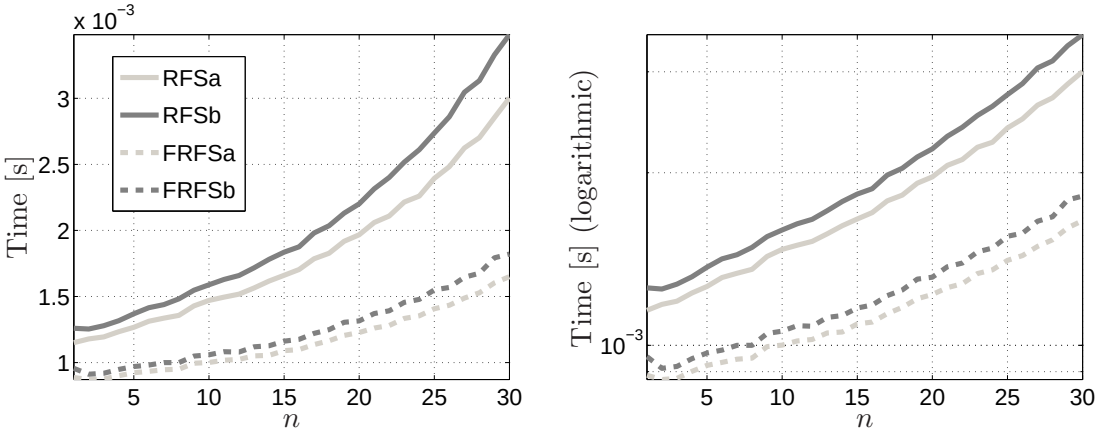


Figure 4.3: Computation time per single recursion with increasing model order n (linear and logarithmic scale).

the Gauss-Newton approach (FRFSa) for the determination of $\sigma_{\tilde{u}}$ appears to be slightly faster than the YW-lin-fast algorithm (FRFSb). The fact that the slope of the curves corresponding to the FRFS algorithms is smaller than those of the RFS approaches illustrates that the computational complexity is reduced from cubic to quadratic order; this underpins the theoretical results obtained in this section. ■

4.2.7 Summary

The above examples have shown that the fast algorithms are generally able to estimate the parameters of an EIV system. The price for the gain in terms of reduced computational cost would appear to be a slower convergence rate in the case of the YW-GN algorithm and more erratic estimates of $\sigma_{\tilde{u}}$ in the case of the YW-lin algorithm. In addition, the fast computation of $\sigma_{\tilde{y}}$ is accompanied with a deterioration of the Frisch-

Name	Determination of			flops
	θ	$\sigma_{\tilde{y}}$	$\sigma_{\tilde{u}}$	
RFSa RFSb	RBCLS (p. 58)	CG-RQ (p. 61)	YW-GN (p. 66) YW-lin (p. 68)	$O(n_\theta^3)$
FRFSa FRFSb	RBCLS (p. 58)	ARQ (p. 100)	YW-GN-fast (p. 93) YW-lin-fast (p. 98)	$O(n_\theta^2)$

Table 4.4: Comparison of RFS and FRFS algorithms (page numbers indicate where the algorithms may be found).

character. Alternative algorithmic combinations seem also feasible, which might be able to reduce or overcome the reduction of the Frisch-character. Improved versions of the FRFS algorithms is therefore a potential topic of further work. Table 4.4 compares the four developed recursive Frisch scheme algorithms for the white noise case.

4.3 Recursive Frisch scheme for coloured output noise

The dynamic Frisch scheme presented in (Beghelli et al. 1990, Söderström 2007*a*) assumes that the additive disturbances on the system input and output are white. Such an assumption, however, can be rather restrictive since the output noise often not solely consists of measurement uncertainties, but also aims to account for process disturbances, which are usually correlated in time. In order to overcome this shortcoming, the Frisch scheme has recently been extended to the coloured output noise case (Söderström 2006, Söderström 2008), i.e. the values of the random process $\{\tilde{y}_i\}_{i=1}^k$ are considered to be correlated in time (see Section 2.4.4 for a detailed review of the offline scheme). This section develops a recursive (adaptive) formulation of the Frisch scheme for coloured output noise, which allows the estimates to be calculated online as new data arrives.

Firstly, a recursive algorithm for the offline scheme, which has been developed in (Söderström 2006) is developed in Section 4.3.1 by making use of two separate Newton algorithms. Whilst the evaluation of the first and second order derivatives is rather computationally demanding, a second algorithm is proposed in Section 4.3.3, which is based on the development in (Söderström 2008). There, some further simplifications have been exploited for the offline scheme, which allows a bilinear parametrisation approach (Ljung 1999, p. 335) to be applied for the corresponding recursive scheme, leading to a computationally more attractive algorithm.

The following assumptions are stated.

AS1 The dynamic system is asymptotically stable, i.e. $A(q^{-1})$ has all zeros inside the unit circle.

AS2 All system modes are observable and controllable, i.e. $A(q^{-1})$ and $B(q^{-1})$ have no common factors.

AS3 The polynomial degrees n_a and n_b are known a priori with $n_b \leq n_a$.

AI1 The true input u_{0_k} is a zero-mean ergodic process and is persistently exciting of sufficiently high order.

AN1a The sequence $\tilde{u}_k$ is a zero-mean, ergodic, white noise process with unknown variance $\sigma_{\tilde{u}}$.

AN1b The sequence $\tilde{y}_k$ is a zero-mean, ergodic noise process with unknown auto-covariance sequence $\{r_{\tilde{y}}(0), r_{\tilde{y}}(1), \cdots\}$.

AN2 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are mutually uncorrelated and also uncorrelated with both u_{0_k} and y_{0_k}.

AE2 The dimension of the instrument vector ζ_k is $n_a + n_b + 1$.

Note that Assumption **AE2** is introduced for convenience only.

4.3.1 Newton algorithm based approach

In this section a recursive algorithm for the identification of dynamical linear errors-in-variables (EIV) systems in the case of coloured output noise, as reviewed in Section 2.4.4, is developed. The input measurement noise variance as well as the auto-covariance elements of the coloured output noise sequence are determined via two separate Newton algorithms. In a similar manner to the recursive bias compensating least squares (RBCLS) approach (cf. Section 3.3.2), the model parameter estimates are obtained by a recursive bias-compensating instrumental variables algorithm with past noisy inputs as instruments, thus allowing the compensation for the explicitly computed bias at each discrete-time instance. The performance of the developed algorithm is demonstrated via simulation. This section is based on the development in (Linden & Burnham 2008*a*), which, in turn, is based on the two-step offline procedure proposed in (Söderström 2006).

Step 1

Recall from Section 2.4.4 that the first step of the Frisch scheme for coloured output noise (FSCON) (see Algorithm 2.1 on page 35) consists of obtaining an estimate of θ and $\sigma_{\tilde{u}}$ by solving the nonlinear least squares (NLS) problem (2.73)

$$\{\hat{\theta}_k, \hat{\sigma}_{\tilde{u}}^k\} = \arg\min_{\theta, \sigma_{\tilde{u}}} \left|\left| G_k\theta - \hat{\xi}_{\delta y}^k \right|\right|^2, \tag{4.38}$$

where G_k is defined in (2.72). By making use of the variable projection principle (see Section 2.3.2), an estimate for $\sigma_{\tilde{u}}$ can be obtained via (cf. (2.75)-(2.76))

$$\hat{\sigma}_{\tilde{u}}^k = \arg\min_{\sigma_{\tilde{u}}} V_1^k \tag{4.39}$$

where

$$V_1^k = \hat{\xi}_{\delta y}^{kT} \hat{\xi}_{\delta y}^k - \hat{\xi}_{\delta y}^{kT} G_k \left[G_k^T G_k\right]^{-1} G_k^T \hat{\xi}_{\delta y}^k. \tag{4.40}$$

In order to obtain a recursive scheme, a basic approach could be to obtain the first and second order derivatives of V_1^k with respect to $\sigma_{\tilde{u}}$ and then to apply an iterative Newton method, where it is iterated once as new data arrives. First, however, in order to satisfy the requirements of a recursive algorithm to store all data in a finite dimensional vector, the covariance matrices are updated via (3.5), i.e. (see Appendix B for more details)

$$\hat{\Sigma}_{\bar{\varphi}}^k = \hat{\Sigma}_{\bar{\varphi}}^{k-1} + \frac{1}{k}\left(\bar{\varphi}_k \bar{\varphi}_k^T - \hat{\Sigma}_{\bar{\varphi}}^{k-1}\right), \tag{4.41a}$$

$$\hat{\Sigma}_{\zeta\bar{\varphi}}^k = \hat{\Sigma}_{\zeta\bar{\varphi}}^{k-1} + \frac{1}{k}\left(\zeta_k \bar{\varphi}_k^T - \hat{\Sigma}_{\zeta\bar{\varphi}}^{k-1}\right). \tag{4.41b}$$

from which the required block matrices are readily obtained.

Recursive Update of $\hat{\sigma}_{\tilde{u}}^k$ For the determination of $\hat{\sigma}_{\tilde{u}}^k$, an iterative optimisation procedure can be utilised to minimise (4.40) where it is iterated once at each step, leading to a recursive scheme (Ljung & Söderström 1983, Ljung 1999). Whilst an iterative Newton method is utilised for this purpose here, it is noted that other choices are also possible. The (undamped) Newton method given by (Ljung 1999, p. 326) is

$$\hat{\sigma}_{\tilde{u}}^k = \hat{\sigma}_{\tilde{u}}^{k-1} - \left[V_1^{k\prime\prime}\right]^{-1} V_1^{k\prime}, \tag{4.42}$$

where $V_1^{k\prime}$ and $V_1^{k\prime\prime}$ denote, respectively, the first and second order derivative of V_k with respect to $\hat{\sigma}_{\tilde{u}}^k$ evaluated at $\hat{\sigma}_{\tilde{u}}^{k-1}$. Explicit expressions for these derivatives are given in Appendices D.1 and D.2, respectively.

Remark 4.2. In order to stabilise the algorithm, it might be advantageous to restrict the search for the input measurement noise variance to the interval

$$0 \leq \sigma_{\tilde{u}} \leq \sigma_{\tilde{u}}^{\max}, \tag{4.43}$$

where $\sigma_{\tilde{u}}^{\max}$ is the maximal admissible value for $\sigma_{\tilde{u}}$, which can be computed from the data as discussed in (Beghelli et al. 1990). Alternatively, a positive constant can be chosen for the maximum admissible value, if such a priori knowledge is available.

Remark 4.3. The derivatives given in Appendices D.1 and D.2 are computed in a straightforward manner. They do not, however, exploit the special structure of the

variable projection problem. The efficient computation of derivatives of orthogonal projectors and pseudoinverses is discussed in (Golub & Pereyra 1973). This approach is, however, not further followed within this thesis.

Recursive Update of $\hat{\theta}_k$ In order to obtain a recursive expression for $\hat{\theta}_k$, a variation of the RBCLS approach given in Section 2.4.1 is used, where the bias of the recursive least squares (RLS) estimate is compensated at each time step k (see also Zheng & Feng 1989, Zheng 1998, Ding et al. 2006).

Recall that the overdetermined IV normal equations (2.69) are given by

$$(\Sigma_{\delta\varphi} - \sigma_{\tilde{u}} J)\, \theta_k = \xi_{\delta y}. \tag{4.44}$$

Whilst an application of the RBCLS principle to an overdetermined set of normal equations is developed in Section 5.3.3, here, the problem is modified to allow for a direct application of the RBCLS principle. Therefore, only the first $n_a + n_b$ equations of (4.44) are considered, since one unknown, namely $\hat{\sigma}_{\tilde{u}}^k$, has already been computed from the set of equations. Disregarding the last equation of (4.44) leads to the definition of a truncated instrument vector

$$\delta_k^* \triangleq \begin{bmatrix} \varphi_{u_k}^T & \zeta_k^{*T} \end{bmatrix}^T = \begin{bmatrix} u_{k-1} & \dots & u_{k-n_b} & u_{k-n_b-1} & \dots & u_{k-n_b-n_\zeta-1} \end{bmatrix}^T. \tag{4.45}$$

The corresponding uncompensated IV estimate is given as

$$\hat{\theta}_k^{\mathrm{IV}} = \Sigma_{\delta^*\varphi}^{-1} \xi_{\delta^* y}, \tag{4.46}$$

which can be recursively computed via a recursive IV (RIV) algorithm (Ljung 1999, p. 369) given by

$$\hat{\theta}_k^{\mathrm{IV}} = \hat{\theta}_{k-1}^{\mathrm{IV}} + L_k \left[y_k - \varphi_k^T \hat{\theta}_{k-1}^{\mathrm{IV}} \right], \tag{4.47a}$$

$$L_k = \frac{P_{k-1}\delta_k^*}{\frac{1-\gamma_k}{\gamma_k} + \varphi_k^T P_{k-1} \delta_k^*}, \tag{4.47b}$$

$$P_k = \frac{1}{1-\gamma_k} \left[P_{k-1} - \frac{P_{k-1}\delta_k^* \varphi_k^T P_{k-1}}{\frac{1-\gamma_k}{\gamma_k} + \varphi_k^T P_{k-1} \delta_k^*} \right], \tag{4.47c}$$

where P_k is scaled such that

$$\left[\hat{\Sigma}_{\delta\varphi}^k \right]^{-1} = P_k. \tag{4.48}$$

This avoids the matrix inversion in (4.46). Following the derivation for the RBCLS (see Section 2.4.1), the recursive bias compensation update equation for $\hat{\theta}_k$ is given by

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{IV}} + \hat{\sigma}_{\tilde{u}}^k P_k J^* \hat{\theta}_{k-1} \tag{4.49}$$

where J^* is obtained by deleting the last row of J. A more detailed derivation can also be found in Appendix E.

Step 2

Recall that the second step of the offline Frisch scheme algorithm for coloured output noise yields an estimate of the autocorrelation sequence of the output noise via (2.81)

$$\hat{\rho}_{\tilde{y}}^k = \alpha_k N(H_k) + H_k^\dagger h_k, \tag{4.50}$$

where H_k and h_k are defined via (2.80) and $N(\cdot)$ denotes the nullspace whilst the scaling factor α_k is required to be determined. In the offline case, this is achieved by solving (cf. (2.84)-(2.85))

$$\hat{\alpha}_k = \arg \min_{\alpha_k} V_2^k, \tag{4.51}$$

where

$$V_2^k = \left(\hat{\sigma}_{\tilde{u}}^k - \hat{\varsigma}_k\right)^2. \tag{4.52}$$

Here, $\hat{\varsigma}_k$ denotes the input measurement noise variance which would be obtained when the $\lambda_{\min}$ equation is utilised, i.e. (see (2.82)-(2.83))

$$\hat{\varsigma}_k \triangleq \lambda_{\min}\left(B_k\left(\alpha_k\right)\right), \tag{4.53}$$

where

$$B_k(\alpha_k) \triangleq \Sigma_{\varphi_u} - \Sigma_{\varphi_u \bar{\varphi}_y}\left[\Sigma_{\bar{\varphi}_y} - \Sigma_{\tilde{\bar{\varphi}}_y}(\alpha_k)\right]^{-1} \Sigma_{\varphi_y \varphi_u}. \tag{4.54}$$

In order to solve (4.51) recursively, a further approximate Newton method is applied where it is iterated once as new data arrives. Consequently, the first and second order derivatives of the cost function V_2^k in (2.85) are to be determined with respect to α_k, which are denoted $V_2^{k\prime}$ and $V_2^{k\prime\prime}$, respectively. In a similar manner as for the minimisation of the YW cost function in Section 3.3.4, the linearised $\lambda_{\min}$ equation is utilised to approximate the first and second order derivatives. These are given by

$$V_2^{k\prime} = -2\left(\hat{\sigma}_{\tilde{u}}^k - \hat{\varsigma}_k\right)\hat{\varsigma}_k', \tag{4.55a}$$

$$V_2^{k\prime\prime} = \hat{\varsigma}_k', \tag{4.55b}$$

where $\hat{\varsigma}_k'$ denotes the derivative of $\hat{\varsigma}_k$ with respect to α_k and for which an approximation is derived in Appendix D.3. The recursive update for $\hat{\alpha}_k$ is therefore given by

$$\hat{\alpha}_k = \hat{\alpha}_{k-1} - \left[V_2^{k\prime\prime}\right]^{-1} V_2^{k\prime}, \tag{4.56}$$

from which the sequence of auto-covariance elements of the output measurement noise $\tilde{y}_k$ is determined via

$$\hat{\rho}_{\tilde{y}}^k = \hat{\alpha}_k N(H_k) + H_k^\dagger h_k. \tag{4.57}$$

The recursive Frisch scheme for correlated output noise, denoted RFSCON1, is summarised as follows.

Algorithm 4.6 (RFSCON1).

$$\hat{\sigma}_{\tilde{u}}^k = \hat{\sigma}_{\tilde{u}}^{k-1} - \left[V_1^{k''}\right]^{-1} V_1^{k'} \tag{4.58a}$$

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{IV}} + \hat{\sigma}_{\tilde{u}}^k P_k J^* \hat{\theta}_{k-1} \tag{4.58b}$$

$$\hat{\theta}_k^{\mathrm{IV}} = \hat{\theta}_{k-1}^{\mathrm{IV}} + L_k \left[y_k - \varphi_k^T \hat{\theta}_{k-1}^{\mathrm{IV}}\right] \tag{4.58c}$$

$$L_k = \frac{P_{k-1}\delta_k^*}{\frac{1-\gamma_k}{\gamma_k} + \varphi_k^T P_{k-1}\delta_k^*}, \tag{4.58d}$$

$$P_k = \frac{1}{1-\gamma_k}\left[P_{k-1} - \frac{P_{k-1}\delta_k^*\varphi_k^T P_{k-1}}{\frac{1-\gamma_k}{\gamma_k} + \varphi_k^T P_{k-1}\delta_k^*}\right] \tag{4.58e}$$

$$\hat{\alpha}_k = \hat{\alpha}_{k-1} - \left[V_2^{k''}\right]^{-1} V_2^{k'} \tag{4.58f}$$

$$\hat{\rho}_{\tilde{y}}^k = \hat{\alpha}_k N(H_k) + H_k^\dagger h_k \tag{4.58g}$$

A more detailed description together with the computational complexity of the RFSCON1 algorithm is given in Table 4.5. It is observed that the overall complexity is of cubic order, which is due the computation of the derivatives of V_1^k and V_2^k as well as due to the determination of the nullspace and pseudo inverse in Step 5.14.

4.3.2 Simulation example

In order to compare the results of the RFSCON1 with the non-recursive algorithm FSCON (cf. Algorithm 2.1 on page 35), a system is chosen similar to that of Example 2 in (Söderström 2008), i.e. a LTI SISO system with $n_a = n_b = 1$, and characterised by

$$\theta = \begin{bmatrix} -0.8 & 2 \end{bmatrix}^T, \qquad \rho_{\tilde{y}} = \begin{bmatrix} 1.96 & 1.37 \end{bmatrix}^T, \qquad \sigma_{\tilde{u}} = 1. \tag{4.59}$$

The values for $r_{\tilde{y}}(0)$ and $r_{\tilde{y}}(1)$ arise by generating the output noise via the auto-regressive model

$$\tilde{y}_k = \frac{1}{1 - 0.7q^{-1}} v_k, \tag{4.60}$$

Step	Description	Procedure	Flops
1	Choose n_ζ, λ_k and j	$n_\zeta = n_a + n_b + 1$, $0 < \lambda_k < 1$, $j = n_\zeta + n_b$	
2	RIV initialisations	$\hat{\theta}^{\text{IV}}_{n_a} = 0$, $P_{n_a} = 0.1I$	
3	Recursion	for $k = n_a + 1, ..., j$	
3.1	Data weighting	$\gamma_k = 1/k$	
3.2	RIV	Compute L_k, $\hat{\theta}^{\text{IV}}_k$ and P_k	
4	General initialisations	$\hat{\Sigma}^j_{\zeta\bar{\varphi}} = 0$, $\hat{\Sigma}^j_{\bar{\varphi}} = \frac{1}{n_\zeta - 1}\sum_{i=n_b+1}^{j} \bar{\varphi}_i \bar{\varphi}_i^T$, $\gamma_j = 1/(n_\zeta - 1)$, $\hat{\sigma}^j_{\tilde{u}} = 0$ and $J^* = \begin{bmatrix} 0 & I_{n_b} \\ 0 & 0 \end{bmatrix}$	
5	Recursion	for $k = j + 1, ...$	
5.1	Update γ	$\gamma_k = \frac{\gamma_{k-1}}{\lambda_k + \gamma_{k-1}}$	2
5.2	Update $\hat{\Sigma}_{\bar{\varphi}}$	$\hat{\Sigma}^k_{\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\bar{\varphi}} + \gamma_k \left(\bar{\varphi}_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\bar{\varphi}} \right)$	$O(n_\theta^2)$
5.3	Update $\hat{\Sigma}_{\zeta\bar{\varphi}}$	$\hat{\Sigma}^k_{\zeta\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}} + \gamma_k \left(\zeta_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}} \right)$	$O(n_\theta^2)$
5.4	RIV estimate	Compute L_k, $\hat{\theta}^{\text{IV}}_k$ and P_k	$O(n_\theta^2)$
5.5	First order derivative	$V_1^{k'}$, see Appendix D.1	$O(n_\theta^3)$
5.6	Second order derivative	$V_1^{k''}$, see Appendix D.2	$O(n_\theta^3)$
5.7	Update of $\hat{\sigma}^k_{\tilde{u}}$	$\hat{\sigma}^k_{\tilde{u}} = \hat{\sigma}^{k-1}_{\tilde{u}} - \left[V_1^{k''} \right]^{-1} V_1^{k'}$	3
5.8	Projection	Project $0 \leq \hat{\sigma}^k_{\tilde{u}} \leq \sigma^{\max}_{\tilde{u}}$	
5.9	Bias compensation	$\hat{\theta}_k = \hat{\theta}^{\text{IV}}_k + \hat{\sigma}^k_{\tilde{u}} P_k J^* \hat{\theta}_{k-1}$	$O(n_\theta)$
5.10	First order derivative	$V_2^{k'} = -2\left(\hat{\sigma}^k_{\tilde{u}} - \hat{\varsigma}_k\right)\hat{\varsigma}'_k$	3
5.11	Determine $\frac{d}{d\alpha_k}\hat{\rho}^k_y$	$\frac{d}{d\alpha_k}\hat{\rho}^k_y = N(H_k)$	$O(n_a^3)$
5.12	Second order derivative	$V_2^{k''} \approx -\frac{\hat{a}^T_{k-1}}{\hat{b}^T_{k-1}\hat{b}_{k-1}} \frac{d}{d\alpha_k}\hat{\Sigma}^k_{\bar{\varphi}y}\hat{a}_{k-1}$	$O(n_a^2)$
5.13	Compute $\hat{\alpha}_k$	$\hat{\alpha}_k = \hat{\alpha}_{k-1} - \left[V_2^{k''} \right]^{-1} V_2^{k'}$	4
5.14	Determine $\hat{\rho}^k_y$	$\hat{\rho}^k_y = \hat{\alpha}_k N(H_k) + H_k^\dagger h_k$	$O(n_a^3)$
		Overall complexity (dominant parts)	$O(n_\theta^3)$

Table 4.5: RFSCON1 algorithm and its computational complexity.

where v_k is a zero-mean white process with unity variance. The system is simulated for $10,000$ samples using a zero mean, white and Gaussian distributed input signal also being of unity variance. The corresponding signal-to-noise ratio for input and output is given by 10.60dB and 39.12dB, respectively.

Choosing $\lambda = 1$, the results for Step 1 are shown in Figure 4.4. The first subplot shows that the Newton method is able to recursively estimate the input measurement noise variance $\sigma_{\tilde{u}}$. The remaining two subplots compare the RIV solution $\hat{\theta}^{\text{IV}}_k$ of the uncompensated normal equations with the recursively compensated Frisch scheme estimate $\hat{\theta}_k$. As expected, the RIV is biased whilst the RFSCON1 successfully compensates for this.

Figure 4.5 shows the estimates of $\rho_{\tilde{y}}$ obtained in Step 2 for both the RFSCON1 as well as the off-line case FSCON. In contrast to the results obtained in Step 1, the

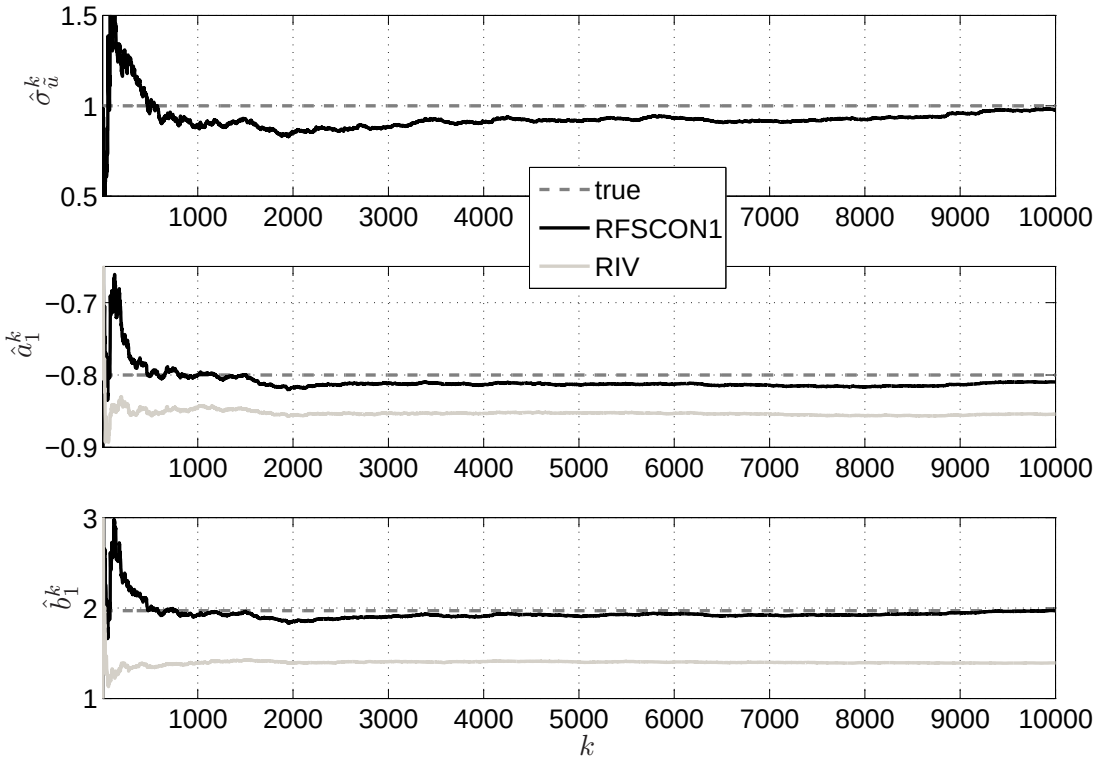


Figure 4.4: Recursive estimates for θ and $\sigma_{\tilde{u}}$ using the RFSCON1 and the biased RIV solution of the uncompensated normal equations.

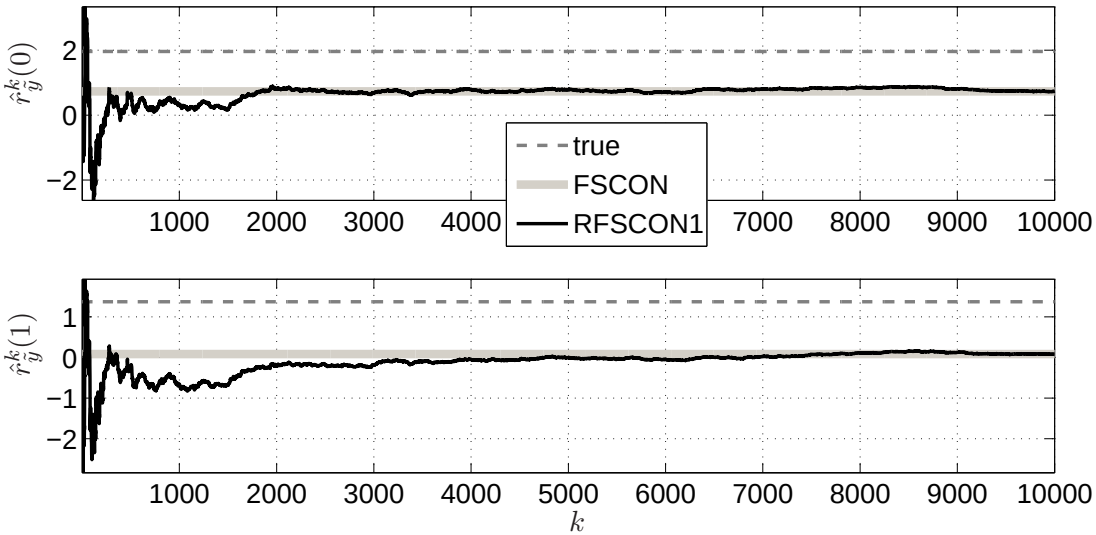


Figure 4.5: Recursive estimates for $r_{\tilde{y}}(0)$ and $r_{\tilde{y}}(1)$ using the RFSCON1 in comparison to the FSCON (offline algorithm).

quality of the estimates obtained in Step 2 for $\hat{\rho}_{\tilde{y}}^k$ would appear to be inferior. This is in agreement with the results reported in (Söderström 2008), where a Monte-Carlo analysis shows poor performance for $\hat{\rho}_{\tilde{y}}^k$ in the non-recursive case. The important observation to note here, however, is that the recursively obtained estimates of $r_{\tilde{y}}(0)$ and $r_{\tilde{y}}(1)$ virtually coincide with their off-line counterparts after $k = 10,000$ recursions. It is also observed that the values of $\hat{r}_{\tilde{y}}^k(0)$ (the estimated variance of the output measurement noise) occasionally exhibit a negative sign during the first 500 recursion steps. This could be avoided by projecting the estimates, such that

$$0 < \hat{\Sigma}_{\bar{\tilde{\varphi}}_y}^k < \hat{\Sigma}_{\bar{\varphi}_y}^k - \hat{\Sigma}_{\bar{\varphi}_y\varphi_u}^k \left[\hat{\Sigma}_{\varphi_u}^k\right]^{-1} \hat{\Sigma}_{\varphi_u\bar{\varphi}_y}^k \tag{4.61}$$

is satisfied (cf. Söderström 2008).

4.3.3 Bilinear parametrisation approach

One of the shortcomings of the RFSCON1 algorithm is its computational complexity. Note that matrix inversions and matrix-matrix multiplications are required to obtain the first and second order derivatives of V_1^k and V_2^k (cf. Appendix D.1 and D.2). In addition, Step 2 can be drastically simplified as outlined in Remark 2.2 (see page 35), which has not yet been exploited.

Step 1

It is worth observing that the NLS problem (4.39) in Step 1 is not only separable for θ and σ, but also bilinear in these variables, i.e. linear in θ for fixed σ and linear in σ for fixed θ. This implies that (2.69), which can be re-expressed as

$$G_k(\sigma_{\tilde{u}})\theta = \xi_{\delta y} \tag{4.62}$$

can also be expressed as

$$S(\theta)\sigma_{\tilde{u}} = s(\theta), \tag{4.63}$$

where $S(\theta) \in \mathbb{R}^{n_b}$ and $s(\theta) \in \mathbb{R}^{n_b}$. Equation (4.63) constitutes an overdetermined set of equations in one unknown, which can be solved by means of LS in a straightforward manner. In order to determine $S(\theta)$ and $s(\theta)$, consider the upper part of (2.68), which can be re-expressed as

$$\left[\Sigma_{\varphi_u\varphi_y} \quad \Sigma_{\varphi_u} - \sigma_{\tilde{u}} I_{n_b}\right] \theta = \xi_{\varphi_u y} \tag{4.64a}$$

$$\Leftrightarrow \qquad \Sigma_{\varphi_u\varphi_y} a + \Sigma_{\varphi_u} b - b\sigma_{\tilde{u}} = \xi_{\varphi_u y} \tag{4.64b}$$

$$\Leftrightarrow \qquad \sigma_{\tilde{u}} b = \Sigma_{\varphi_u\varphi_y} a + \Sigma_{\varphi_u} b - \xi_{\varphi_u y}. \tag{4.64c}$$

By inspection, it is clear that

$$S(\theta) = b, \tag{4.65a}$$

$$s(\theta) = \hat{\Sigma}^k_{\varphi_u\varphi_y} a + \hat{\Sigma}^k_{\varphi_u} b - \hat{\xi}^k_{\varphi_u y}. \tag{4.65b}$$

Consequently, assuming $\hat{\theta}_{k-1}$ is available and sufficiently close to $\hat{\theta}_k$, θ can be replaced by $\hat{\theta}_{k-1}$ which allows $\hat{\sigma}^k_{\tilde{u}}$ to be obtained via the minimum norm solution

$$\hat{\sigma}^k_{\tilde{u}} = S(\hat{\theta}_{k-1})^\dagger s(\hat{\theta}_{k-1}), \tag{4.66}$$

where, assuming $b^T b \neq 0$, the left pseudo inverse of $S(\hat{\theta}_k)$ is given by

$$S(\theta)^\dagger = \frac{b^T}{b^T b}. \tag{4.67}$$

The parameter vector is then obtained as the LS solution $\hat{\theta}_k = G_k^\dagger(\hat{\sigma}^k_{\tilde{u}})\hat{\xi}^k_{\delta y}$. Alternatively, a recursive bias compensation rule as in (4.49) using the RIV algorithm (4.47) could be utilised. However, when use was made of the recursive bias compensation rule, the estimation results appeared to be sensitive in simulation with respect to the initialisation of the algorithm. Consequently, the offline LS solution for $\hat{\theta}_k$ is utilised.

Step 2

As pointed out in Remark 2.2 (cf. page 35), Step 2 simplifies to solving

$$\bar{H}_k(\theta)\rho_{\tilde{y}} = \bar{h}_k(\theta), \tag{4.68}$$

where an explicit expression for $\bar{h}_k$ is given by (Söderström 2008)

$$\bar{h}_k(\theta) = \begin{bmatrix} \hat{\Sigma}^k_{\bar{\varphi}_y} & \hat{\Sigma}^k_{\bar{\varphi}_y\varphi_u} \end{bmatrix} \bar{\theta}, \tag{4.69}$$

whilst $\bar{H}_k(\theta)$ satisfies

$$\bar{H}_k(\theta)\rho_{\tilde{y}} = \Sigma_{\bar{\tilde{\varphi}}_y}\bar{\theta}, \tag{4.70}$$

where the symmetric Toeplitz matrix $\Sigma_{\bar{\tilde{\varphi}}_y}$ is defined by

$$\Sigma_{\bar{\varphi}_y} = \begin{bmatrix} r_{\tilde{y}}(0) & \cdots & r_{\tilde{y}}(n_a) \\ \vdots & \ddots & \vdots \\ r_{\tilde{y}}(n_a) & \cdots & r_{\tilde{y}}(0) \end{bmatrix}. \tag{4.71}$$

Step	Description	Procedure	Flops
1	Choose n_ζ, λ_k and j	$n_\zeta = n_a + n_b + 1$, $0 < \lambda_k < 1$, $j = n_\zeta + n_b$	
2	General initialisations	$\hat{\Sigma}^j_{\zeta\bar{\varphi}} = 0$, $\hat{\Sigma}^j_{\bar{\varphi}} = \frac{1}{n_\zeta - 1}\sum_{i=n_b+1}^{j} \bar{\varphi}_i \bar{\varphi}_i^T$, $\gamma_j = 1/(n_\zeta - 1)$, $\hat{\sigma}^j_{\tilde{u}} = 0$ and $J^* = \begin{bmatrix} 0 & I_{n_b} \\ 0 & 0 \end{bmatrix}$	
3	Recursion	for $k = j+1, \ldots$	
3.1	Update γ	$\gamma_k = \frac{\gamma_{k-1}}{\lambda_k + \gamma_{k-1}}$	2
3.2	Update $\hat{\Sigma}_{\bar{\varphi}}$	$\hat{\Sigma}^k_{\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\bar{\varphi}} + \gamma_k \left(\bar{\varphi}_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\bar{\varphi}} \right)$	$O(n_\theta^2)$
3.3	Update $\hat{\Sigma}_{\zeta\bar{\varphi}}$	$\hat{\Sigma}^k_{\zeta\bar{\varphi}} = \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}} + \gamma_k \left(\zeta_k \bar{\varphi}_k^T - \hat{\Sigma}^{k-1}_{\zeta\bar{\varphi}} \right)$	$O(n_\theta^2)$
3.4	Update of $\hat{\sigma}^k_{\tilde{u}}$	$\hat{\sigma}^k_{\tilde{u}} = \frac{\hat{b}^T_{k-1}}{\hat{b}^T_{k-1}\hat{b}_{k-1}} \left[\hat{\Sigma}^k_{\varphi_u \varphi_y} \hat{a}_{k-1} + \hat{\Sigma}^k_{\varphi_u} \hat{b}_{k-1} - \hat{\xi}^k_{\varphi_u y} \right]$	$O(n_a n_b)$
3.5	Projection	Project $0 \leq \hat{\sigma}^k_{\tilde{u}} \leq \sigma^{\max}_{\tilde{u}}$	
3.6	Compute $\hat{\theta}_k$	$\hat{\theta}_k = G_k^\dagger(\hat{\sigma}^k_{\tilde{u}}) \hat{\xi}^k_{\delta y}$	$O(n_\theta^3)$
3.7	Compute $\hat{\rho}^k_{\tilde{y}}$	$\hat{\rho}^k_{\tilde{y}} = \bar{H}_k^\dagger(\hat{\theta}_k) \bar{h}_k(\hat{\theta}_k)$	$O(n_a^3)$
		Overall complexity (dominant parts)	$O(n_\theta^3)$

Table 4.6: RFSCON2 algorithm and its computational complexity.

Since the matrix $\bar{H}_k(\theta)$ is not updated via an ordinary rank-one update, $\rho_{\tilde{y}}$ is computed here via the offline LS solution as

$$\hat{\rho}^k_{\tilde{y}} = \bar{H}_k^\dagger(\hat{\theta}_k) \bar{h}_k(\hat{\theta}_k). \tag{4.72}$$

The overall algorithm is denoted RFSCON2 and can be summarised as follows.

Algorithm 4.7 (RFSCON2).

$$\hat{\sigma}^k_{\tilde{u}} = \frac{\hat{b}^T_{k-1}}{\hat{b}^T_{k-1}\hat{b}_{k-1}} \left[\hat{\Sigma}^k_{\varphi_u \varphi_y} \hat{a}_{k-1} + \hat{\Sigma}^k_{\varphi_u} \hat{b}_{k-1} - \hat{\xi}^k_{\varphi_u y} \right] \tag{4.73a}$$

$$\hat{\theta}_k = G_k^\dagger(\hat{\sigma}^k_{\tilde{u}}) \hat{\xi}^k_{\delta y} \tag{4.73b}$$

$$\hat{\rho}^k_{\tilde{y}} = \bar{H}_k^\dagger(\hat{\theta}_k) \bar{h}_k(\hat{\theta}_k) \tag{4.73c}$$

A more detailed description together with the computational complexity of the RFSCON2 algorithm is given in Table 4.6. Note that due to the usage of the pseudo inverses, the algorithm is of cubic complexity.

4.3.4 Simulation example

In order to compare the estimates obtained by the RFSCON1 with those obtained by the RFSCON2, the simulation in Section 4.3.2 is repeated and both algorithms are

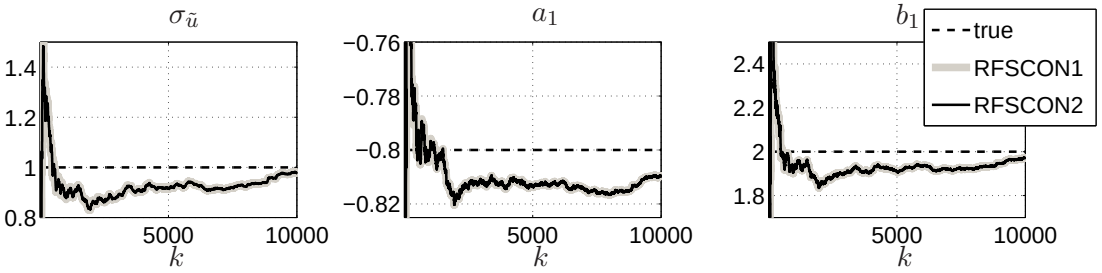


Figure 4.6: Recursive estimates for θ and $\sigma_{\tilde{u}}$ using the RFSCON1 and the RFSCON2.

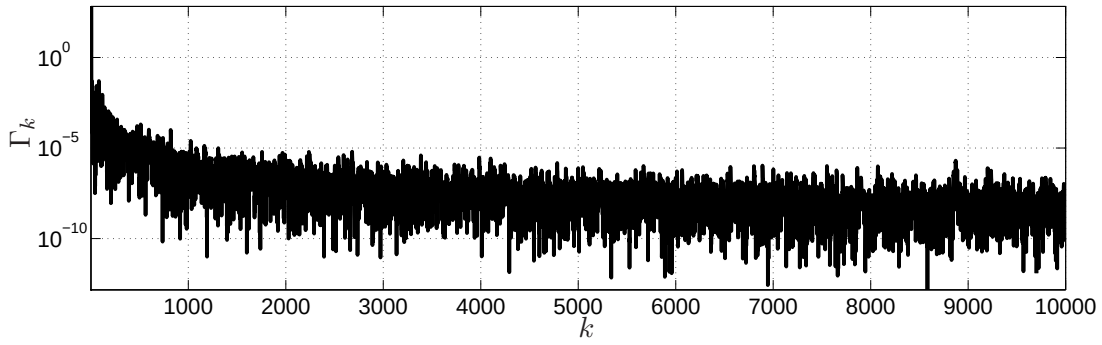


Figure 4.7: Mean square difference as defined in (4.74), between RFSCON1 and RFSCON2 estimates for $\hat{\theta}_k$ and $\hat{\sigma}_{\tilde{u}}^k$.

applied to estimate Θ (cf. (2.62)). The estimates of $\sigma_{\tilde{u}}$ and θ are shown in Figure 4.6. It is observed that the estimates of $\sigma_{\tilde{u}}$ and θ using the RFSCON1 and the RFSCON2 are virtually identical. This is confirmed in Figure 4.7, where the mean square difference between both estimates defined by

$$\Gamma_k \triangleq \frac{1}{n_\theta + 1} \left(\begin{bmatrix} \hat{\theta}_k^1 \\ \hat{\sigma}_{\tilde{u}}^{k,1} \end{bmatrix} - \begin{bmatrix} \hat{\theta}_k^2 \\ \hat{\sigma}_{\tilde{u}}^{k,2} \end{bmatrix} \right)^T \left(\begin{bmatrix} \hat{\theta}_k^1 \\ \hat{\sigma}_{\tilde{u}}^{k,1} \end{bmatrix} - \begin{bmatrix} \hat{\theta}_k^2 \\ \hat{\sigma}_{\tilde{u}}^{k,2} \end{bmatrix} \right), \tag{4.74}$$

is presented. Note that in (4.74) the superscript 1 denotes the estimates of the RFSCON1 and the superscript 2 denotes the estimates of the RFSCON2. Whilst it was expected that both estimators would yield similar results, the fact that the estimates for $\sigma_{\tilde{u}}$ and θ are virtually identical is rather surprising.

The estimates of $\rho_{\tilde{y}}$ are shown in Figure 4.8. It is observed, that the estimates of the auto-covariance sequence of the output measurement noise are different for both algorithms. After 10,000 recursion steps, the estimates of the RFSCON2 even exhibit an incorrect sign. In addition, the estimates obtained by the RFSCON2 would appear to have a higher variance when compared to those obtained by the RFSCON1.

It is also of interest to compare the computation time of both algorithms. For this single first order identification problem, the computation time per recursion could be reduced from approximately $1.6 \cdot 10^{-3}$s in the case of the RFSCON1, to $3.7 \cdot 10^{-4}$s in the case of the RFSCON2. This means that the latter algorithm appears to be four

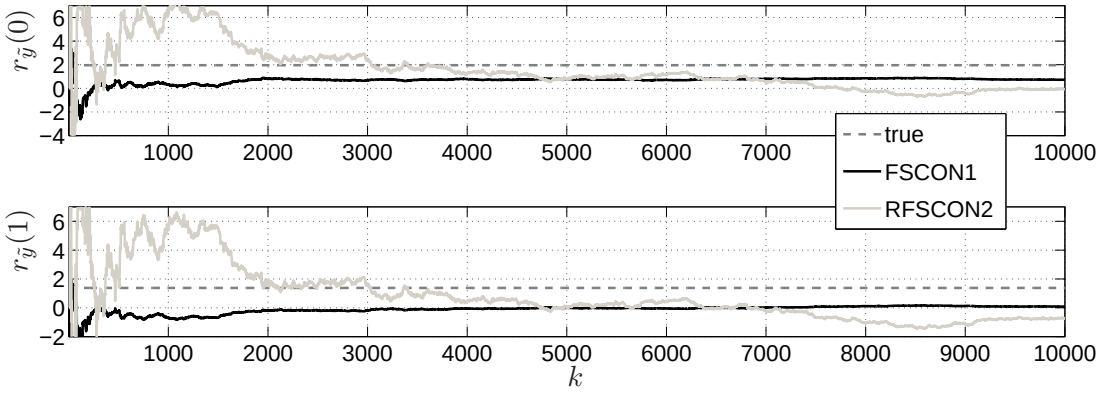


Figure 4.8: Recursive estimates for $r_{\tilde{y}}(0)$ and $r_{\tilde{y}}(1)$ using the RFSCON1 and the RFSCON2.

times faster than the RFSCON1.

4.3.5 Summary & discussion

The RFSCON1 makes use of a Newton algorithm to solve the variable projection subproblem for the determination of the input noise variance, recursively. Another approximate Newton algorithm is applied, in order to determine the auto-covariance sequence of the coloured output noise, where the approximate derivatives are, as in the case of the YW-lin algorithm developed in Section 3.3.4, obtained by considering the linearised Frisch equations. Computing the derivatives, however, is a computationally expensive task and whilst other approaches for the recursive solution of the variable projection problem will be discussed in Chapter 5, the solution leading to the RFSCON2 exploits the bilinear parametrisation structure of the problem. Thus, the variable projection problem is avoided and, by making use of additional simplifications reported in (Söderström 2008), the RFSCON2 appears to be computationally more attractive than its RFSCON1 counterpart. One remaining potential shortcoming of this fast algorithm is, however, that the two LS problems of dimension $n_{\bar{\theta}}$ and $n_{\bar{a}}$, respectively, are to be solved in an offline manner at each recursion step (cf. (4.72)).

The convergence aspects of both algorithms remain a open field of research. Although it is stated in (Ljung 1999, p. 335) that a multi-stage identification procedure exploiting the bilinear parametrisation will lead to a local minima, a more thorough analysis might be required and, as such, is identified as potential further work.

4.4 Concluding remarks

Novel algorithms based on the previously proposed recursive Frisch-YW algorithms have been developed, in order to reduce the computation time per recursion. Whilst the computational complexity of the RFS algorithms, which have been developed in

Chapter 3, is of cubic order, the new approaches, denoted FRFS, only require $O(n_\theta^2)$ flops, where n_θ is the number of model parameters to be identified. Therefore, the fast FRFS algorithms might be considered as being more applicable in practical applications, especially when restrictions on the available computation power become constraining factors. The computational savings have been achieved by introducing additional approximations and by making use of the principle of stationary iterative least squares. The improvement in terms of computation time, however, is accompanied with a reduction of the Frisch-character; a measure of how well the recursive algorithm resembles the unique property the offline Frisch scheme. The estimates of the fast algorithms do, however, appear to be very close to those of the RFS algorithms. Consequently, the FRFS approaches appear to be an attractive alternative to the RFS algorithms, if the Frisch-character of the estimates is not of importance. The additional approximations which are introduced in order to derive the fast algorithm, however, might also effect the convergence properties of the algorithm. It is noted that this requires some more theoretical analysis and has been identified as a potential item for further work.

In the second part of this chapter, two recursive algorithms for the Frisch scheme in the case of coloured output noise have been developed based on their offline counterparts. With respect to practical application, such an extension can be considered to be of major importance, since the output noise usually aims to model measurement noise as well as process disturbances. The first algorithm, which makes repeated use of Newton's method for the minimisation of the two separate cost functions, is rather computationally demanding. The second algorithm is able to overcome this shortcoming by exploiting the special structure of the problem. This reduces the estimation task to three linear least squares problems, which can be solved efficiently at each time step, leading to a recursive identification scheme. Whilst further numerical and theoretical analyses for these algorithms may be required, both schemes appear to work well for the simulation example considered.

The convergence and consistency aspects off all algorithms presented within this Chapter have not yet been analysed. Consequently, there is scope for potential further work.

Chapter 5

Recursive extended bias compensating least squares

Contents

Nomenclature

$f(\sigma)$	Auxiliary vector
$F(\sigma)$	Auxiliary matrix
F_2^k	Measure for Frisch-character
$\bar{F}$	Averaged Frisch-character
$g(\sigma)$	Auxiliary vector
$G(\sigma)$	Auxiliary matrix
$J(\vartheta)$	Jacobian (of residual $r_k(\theta, \sigma)$ with respect to ϑ)

L_k ..... Recursive least squares gain
m ..... Model order
MSE_k ... Mean squared error
$r(\theta, \sigma)$ ... Nonlinear least squares residual
P_k ..... Scaled covariance matrix obtained from recursive least squares
z_k ..... Instrument vector
γ_k ..... Scaling factor, step size
ζ_k ..... Instrument vector
$\hat{\theta}_k^{\mathrm{IV}}$ ..... Uncompensated instrumental variable estimate
$\hat{\theta}_{k+\frac{1}{2}}$ .... Intermediate estimate of θ
$\hat{\psi}_k$ ..... Gauss-Newton update direction corresponding to ϑ
$\hat{\psi}_k^{\theta}$ ..... Gauss-Newton update direction corresponding to θ
$\hat{\psi}_k^{\theta}$ ..... Gauss-Newton update direction corresponding to σ

Preliminary reading: Sections 2.2, 2.3, 2.4.2.

5.1 Introduction

The Frisch scheme which utilises the Yule-Walker (YW) model selection criterion (Frisch-YW) essentially involves the solution of a set of nonlinear equations. These consist of the bias compensated normal equations, the $\lambda_{\min}$-equation as well as the high-order YW equations. Recent developments (Hong et al. 2007, Hong & Söderström 2008) have shown, that the same set of nonlinear Frisch-YW equations can also be obtained by considering an extended bias compensating least squares (EBCLS) approach (cf. Section 2.4.2). The EBCLS method (Ekman 2005) is another recently developed EIV identification technique, which allows the estimation of the model parameters as well as the measurement noise variances by solving a nonlinear least squares (NLS) problem. Rather than utilising the least squares (LS) normal equations, as used for the determination of the system parameter vector within the bias compensating least squares technique (BCLS), it makes use of the instrumental variable (IV) approach to obtain a set of overdetermined normal equations. The number and the type of instruments are chosen such that there are sufficient equations to determine not only the parameter vector, but also the measurement noise variances. By choosing the instruments in a particular way, the set of nonlinear Frisch-YW equations is obtained. Whilst the underlying objective is still focused on the development of recursive Frisch scheme algorithms, this chapter investigates approaches within the EBCLS framework, rather than following a direct approach as in Chapters 3-4. The developed recursive algorithms, however, are not restricted to the particular choice of instruments, which lead to the Frisch-YW, but may also be applied to more general EBCLS cases. In particular, since the Frisch-EM case (cf. Section 2.4.3) can also be interpreted as an EBCLS problem, the results in this chapter could also be extended to this case. In contrast, a connection between the Frisch-CM and the EBCLS is not known.

Section 5.2 states the particular EBCLS setup, which is equivalent to the Frisch-

YW problem. Subsequently, recursive algorithms based on the bilinear parametrisation principle are developed in Section 5.3, whilst Section 5.4 considers recursive variable projection algorithms. Section 5.5 provides an extensive simulation study, which not only analyses the algorithms developed within this chapter, but also provides a comparison with the recursive Frisch scheme algorithm from Chapters 3 and 4. Concluding remarks are given in Section 5.6.

Part of the material presented within this chapter has been published in (Linden & Burnham 2008*b*).

5.2 Equivalent EBCLS representation

It has been shown in (Hong et al. 2007, Hong & Söderström 2008) that it is possible to reformulate the Frisch-YW equations (3.1) as an EBCLS problem. The equations which are used within the Frisch-YW scheme can be summarised as

$$\xi_{y\varphi}\theta = r_y(0) - \sigma_{\tilde{y}}, \tag{5.1a}$$

$$\left[\Sigma_\varphi - \Sigma_{\tilde{\varphi}}(\sigma)\right]\theta = \xi_{\varphi y}, \tag{5.1b}$$

$$\Sigma_{\zeta\varphi}\theta = \xi_{\zeta y}, \tag{5.1c}$$

where $\sigma = [\sigma_{\tilde{y}}\ \sigma_{\tilde{u}}]^T$ and the remaining entities are defined by (2.10)-(2.12). Indeed, this corresponds to the EBCLS framework (see Section 2.4.2) with the particular choice of instruments given by

$$z_k = \begin{bmatrix} y_k & \varphi_k^T & \zeta_k^T \end{bmatrix}^T \in \mathbb{R}^{n_z} \tag{5.2}$$

where φ_k is the regression vector defined by (2.6c) whilst ζ_k is given by (2.50).

Remark 5.1 (Frisch-character of the EBCLS solution). The Frisch scheme equations (5.1) are equivalent to those used in Chapter 3 given by (3.1). The actual difference between both representations is that the latter forces (3.1a) to hold exactly, which is achieved by computing $\hat{\sigma}_{\tilde{y}}^k$ via the $\lambda_{\min}$ equation (3.1b). This ensures the Frisch-character of the solution, or stated differently, the singularity of $\Sigma_{\bar{\varphi}} - \Sigma_{\tilde{\bar{\varphi}}}$. Within the EBCLS framework, this would imply that the estimate obtained via the NLS problem (5.1) is solved in a way, such that (5.1a)-(5.1b) is exactly satisfied, whilst (5.1c), which basically corresponds to the equations of the YW model selection criterion, needs only to hold approximately. Whilst this might be achieved by an appropriate weighting scheme for Equations (5.1), this is not considered within this thesis. Consequently, it cannot be expected that the solution of the EBCLS problem will satisfy the Frisch-character. This will be further investigated in the numerical example given in Section 5.5.

Following the interpretation within the EBCLS framework, the Frisch-YW equations (5.1) simplify to (cf. (2.33))

$$[\Sigma_{z\varphi} - \Sigma_{\tilde{z}\tilde{\varphi}}(\sigma)]\,\theta = \xi_{zy} - \xi_{\tilde{z}\tilde{y}}(\sigma), \tag{5.3}$$

where the compensating matrices (noting that $n_z = n_\theta + n_\zeta + 1$) of this particular white noise Frisch-YW case are given by

$$\Sigma_{\tilde{z}\tilde{\varphi}}(\sigma) = \begin{bmatrix} 0 & 0 \\ \sigma_{\tilde{y}} I_{n_a} & 0 \\ 0 & \sigma_{\tilde{u}} I_{n_b} \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{n_z \times n_\theta}, \tag{5.4a}$$

$$\xi_{\tilde{z}\tilde{y}}(\sigma) = \begin{bmatrix} \sigma_{\tilde{y}} \\ 0 \\ 0 \end{bmatrix}. \tag{5.4b}$$

Utilising the notation used for the variable projection problem introduced in Section 2.3.2, the residual to be minimised is given by

$$r(\theta, \sigma) = F(\sigma)\theta - f(\sigma), \tag{5.5}$$

where $F(\sigma)$ and $f(\sigma)$ are given by (2.35a)

$$F(\sigma) = \Sigma_{z\varphi} - \Sigma_{\tilde{z}\tilde{\varphi}}(\sigma), \tag{5.6a}$$

$$f(\sigma) = \xi_{zy} - \xi_{\tilde{z}\tilde{y}}(\sigma). \tag{5.6b}$$

In contrast to the more general EBCLS cases, where the input and output measurement noises may be coloured, the resulting equations for the white noise Frisch-YW case are not only linear in θ, but also linear in σ. This means, that (5.5) may be equivalently expressed as

$$r(\theta, \sigma) = G(\theta)\sigma - g(\theta), \tag{5.7}$$

where it is straightforward to verify that

$$G(\theta) = \begin{bmatrix} 1 & 0 \\ -a & 0 \\ 0 & -b \\ 0 & 0 \end{bmatrix}, \tag{5.8a}$$

$$g(\theta) = \xi_{zy} - \Sigma_{z\varphi}\theta. \tag{5.8b}$$

Such bilinear parametrisations are conceptually simpler than the more general case with $r(\theta, \sigma)$ being separable in θ and σ. Whilst the latter scenario can effectively be tackled via the variable projection approach as discussed in Section 2.3.2, the problem in the bilinear parametrisation case reduces to a two-stage linear LS problem which can be solved efficiently in an iterative/recursive manner (cf. p. 335 Ljung 1999).

As a first approach, a recursive two-step algorithm, which takes advantage of the bilinear parametrisation, is developed. In order to reduce the complexity from cubic to quadratic order, the recursive BCLS technique, which has already been used in Chapter 3 for the implementation of the RFS algorithms (cf. Section 3.3.2), can be applied. However, the recursive BCLS approach has to be modified in order to deal with the set of overdetermined normal equations. A second approach considers the recursive implementation of the more general variable projection method, for which a Gauss-Newton algorithm is considered. Whilst such an approach might not be required for the bilinear parametrisation case (as in the Frisch-YW case), it is of interest since it can easily be extended to deal with more general EBCLS cases when $r(\theta, \sigma)$ is not linear in σ.

As for the RFS approaches of Chapter 3, the following assumptions are stated.

AS1 The dynamic system is asymptotically stable, i.e. $A(q^{-1})$ has all zeros inside the unit circle.

AS2 All system modes are observable and controllable, i.e. $A(q^{-1})$ and $B(q^{-1})$ have no common factors.

AS3 The polynomial degrees n_a and n_b are known a priori with $n_b \leq n_a$.

AI1 The true input u_{0_k} is a zero-mean ergodic process and is persistently exciting of sufficiently high order.

AN1 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are zero-mean, ergodic, white noises with unknown variances, denoted $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, respectively, i.e.

$$\sigma_{\tilde{u}} \delta_{kl} \triangleq E\left[\tilde{u}_k \tilde{u}_l\right], \tag{5.9a}$$

$$\sigma_{\tilde{y}} \delta_{kl} \triangleq E\left[\tilde{y}_k \tilde{y}_l\right]. \tag{5.9b}$$

AN2 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are mutually uncorrelated and also uncorrelated with both u_{0_k} and y_{0_k}.

5.3 Bilinear parametrisation

This section considers a two-stage algorithm, in order to recursively solve the Frisch-YW equations, by exploiting the bilinear parametrisation. The two resulting linear

least squares problems

$$F_k(\hat{\sigma}_{k-1})\hat{\theta}_k = f_k(\hat{\sigma}_{k-1}), \tag{5.10a}$$
$$G(\hat{\theta}_k)\hat{\sigma}_k = g_k(\hat{\theta}_k), \tag{5.10b}$$

are solved alternately at each discrete-time instance k, where

$$F_k(\hat{\sigma}_{k-1}) = \hat{\Sigma}^k_{z\varphi} - \hat{\Sigma}^k_{\tilde{z}\tilde{\varphi}}(\hat{\sigma}_{k-1}), \tag{5.11a}$$
$$f_k(\hat{\sigma}_{k-1}) = \hat{\xi}^k_{zy} - \hat{\xi}^k_{\tilde{z}\tilde{y}}(\hat{\sigma}_{k-1}), \tag{5.11b}$$
$$G(\hat{\theta}_k) = \begin{bmatrix} 1 & -\hat{a}_k^T & 0 & 0 \\ 0 & 0 & -\hat{b}_k^T & 0 \end{bmatrix}^T, \tag{5.11c}$$
$$g_k(\hat{\theta}_k) = \hat{\xi}^k_{zy} - \hat{\Sigma}^k_{z\varphi}\hat{\theta}_k. \tag{5.11d}$$

Note that the quantities $F_k(\hat{\sigma}_{k-1})$, $f_k(\hat{\sigma}_{k-1})$, $G(\hat{\theta}_k)$ and $g_k(\hat{\theta}_k)$ are time dependent either due to the incorporated covariance elements which are updated as new data arrives and/or since the current estimates of σ and θ are updated at each recursion[1].

Whilst the covariance elements can be updated in the usual manner (cf. Appendix B) via

$$\hat{\Sigma}^k_{z\varphi} = (1-\gamma_k)\hat{\Sigma}^{k-1}_{z\varphi} + \gamma_k z_k \varphi_k^T, \tag{5.12a}$$
$$\hat{\xi}^k_{zy} = (1-\gamma_k)\hat{\xi}^{k-1}_{z\varphi} + \gamma_k z_k y_k, \tag{5.12b}$$

where γ_k is a scaling factor, the identification problem essentially reduces to the solution of two sets of linear equations (5.10) at each time instance. Consider equation (5.10b) first. Due to the sparse structure of $G(\hat{\theta}_k)$ the left pseudo inverse can be computed 'cheaply' and is explicitly given by (see. Appendix F)

$$G^\dagger(\hat{\theta}_k) = \begin{bmatrix} \frac{1}{1+\hat{a}_k^T\hat{a}_k} & \frac{-\hat{a}_k^T}{1+\hat{a}_k^T\hat{a}_k} & 0 & 0 \\ 0 & 0 & \frac{-\hat{b}_k^T}{\hat{b}_k^T\hat{b}_k} & 0 \end{bmatrix}. \tag{5.13}$$

Note that only vector-vector multiplications are required, which is of linear complexity only (cf. Table 2.1 on page 21). Thus, (5.10b) can be solved in a straightforward linear LS manner and a direct computation of $\hat{\sigma}_k$ is given by

$$\hat{\sigma}_k = G_k^\dagger(\hat{\theta}_k)g(\hat{\theta}_k). \tag{5.14}$$

In order to obtain a solution of the overdetermined set of normal equations (5.10a), several options are discussed in the following subsection.

[1]The quantity G does not require a subscript k, since it does not contain any covariance elements.

5.3.1 Algorithms for overdetermined normal equations

In order to obtain $\hat{\theta}_k$, an overdetermined set of normal equations given by (5.10a) is required to be solved at each time instance. Several options are available and five such possibilities are outlined below.

1. **(LS)** The LS method can be applied to solve (5.10a) in an offline manner at each recursion, yielding the minimum norm solution

$$\hat{\theta}_k = F_k^{\dagger}(\hat{\sigma}_{k-1}) f_k(\hat{\sigma}_{k-1}). \tag{5.15}$$

   An advantage of this approach is that it leads to a robust implementation by making use of well known and readily implemented routines (Lawson & Hanson 1995). Such an approach is feasible since the dimension of the problem is fixed, i.e. the number of rows in (5.10a) do not grow with time. A potential drawback could be that the computational costs could be too large for an online implementation. The computational complexity of the LS algorithm is generally of cubic order, which might forbid a practical implementation if the number of instruments n_z is large. This does, however, depend on the particular application, more specifically the number of instruments, the sampling interval as well as the available hardware.

2. **(TLS)** The total least squares (TLS) method could be applied to estimate θ. Using a Matlab-like notation (MathWorks 2007), the TLS estimate of θ is given by

$$\hat{\theta}_k = \frac{1}{V_{1,n_\theta+1}} V_{2:n_\theta+1,n_\theta+1}, \tag{5.16}$$

   where V is the matrix of right singular vectors obtained via the singular value decomposition

$$\begin{bmatrix} f_k(\hat{\sigma}_{k-1}) & F_k(\hat{\sigma}_{k-1}) \end{bmatrix} = USV^T. \tag{5.17}$$

   In (5.16), the quantity $V_{1,n_\theta+1}$ denotes the first element of the last column of V whilst $V_{2:n_\theta+1,n_\theta+1}$ denotes the last column of V starting from 2 up to the last row $n_\theta + 1$. The estimates of θ obtained via the LS and TLS differ only for short sample lengths (i.e. for a finite number of observations), but their asymptotic accuracy is identical[2] as pointed out in (Söderström & Mahata 2002). There it is concluded that the LS estimator appears to be more robust, whilst the TLS estimator is computationally less demanding (Van Huffel & Vandewalle 1991).

3. **(RLS)** A direct application of the RLS algorithm would certainly be desirable,

[2]Note that this applies only for the overdetermined system of normal equations considered here. For a general linear systems of equations it is not necessarily true that the LS and TLS estimates are asymptotically equivalent.

but it is not straightforward. This is due to the fact that the matrix $F_k(\hat{\sigma}_{k-1})$ is not updated via a rank-one update, but $F_k(\hat{\sigma}_{k-1}) - F_{k-1}(\hat{\sigma}_{k-2})$ is generally of full rank. This, coupled with the fact that the set of normal equations (5.10a) is overdetermined, prevents the application of the matrix inversion lemma to give a straightforward recursive algorithm. Note that recursive schemes based on matrix factorisation updates (cf. Point 2 on page 86) also rely on the rank-one update, hence are also not straightforward to apply.

4. **(RBCLS)** It is possible to make use of a more general form of the recursive bias compensating least squares (RBCLS) procedure (see Section 3.3.2 and Appendix A), which is essentially based on the stationary iterative LS principle (cf. Section 2.3.1). This means, that at each time instance k, the (uncompensated, hence biased) IV estimate is calculated as

$$\hat{\Sigma}_{z\varphi}^{k}\hat{\theta}_{k}^{\mathrm{IV}} = \hat{\xi}_{zy}^{k}. \tag{5.18}$$

Subsequently, the bias, which can be determined based on the current estimate of input and output measurement noise variances, is removed at each time step k. In order to recursively solve the overdetermined system of normal equations (5.18), two different approaches of extended recursive least squares (ERLS) are possible:

a) **(ERLS1)** One way to deal with an overdetermined system of equations has been discussed in (Feng, Zhang, Zhang & Bao 2001). There, an extension of the matrix inversion lemma has been proposed which can be utilised to derive a recursive estimator[3] for the overdetermined normal equations (5.18). Such an algorithm, which is denoted ERLS1 (extended recursive least squares), is considered in Section 5.3.2.

b) **(ERLS2)** A recursive algorithm for the overdetermined set of normal equations based on the (standard) matrix inversion lemma is given in (Friedlander 1984, Söderström & Stoica 1989). There, the problem is reformulated, such that only the inversion of a 2×2 matrix is required. A detailed description of the algorithm, which is here denoted ERLS2, is given in Appendix G.2.

Note that the usage of the RBCLS technique can cause divergence of the overall algorithm as discussed in Section 3.6.4. Therefore, it remains to be evaluated if the reduction in computational complexity is worth exploiting given the potential deterioration of the convergence properties of the algorithm. This aspect is further investigated in Section 5.5.

5. **(Alternative iterative solvers)** If n_z is considered to be too large for a repeated application of batch LS techniques and if a RBCLS approach is also not desired,

[3] See also Section 5.3.5 below.

an alternative might be to use iterative solvers (Björck 1996, Ch. 7), such as the conjugate gradient method for linear LS problems, in a recursive fashion. Such an approach has been considered in (Chang & Wilson 2000, Björck 1997), but is not further investigated within this thesis.

In the development of this chapter, the usage of the offline LS (Point 1) as well as the RBCLS approaches (Point 4a) are considered.

5.3.2 Least squares based approach

Motivated by the foregoing discussion, the first algorithm presented in this section solves the LS problems (5.10) in an offline manner at each time instance k and will be utilised for future comparison purposes. The algorithm is denoted RBP1 (recursive bilinear parametrisation) and is summarised as follows:

Algorithm 5.1 (RBP1).

1) Set $j = n_\zeta + n_b$, $\hat{\sigma}_j = 0$

2) For $k = j + 1, ...$

a) Update $\hat{\Sigma}_{z\varphi}^k$ and $\hat{\xi}_{zy}^k$ via (5.12)

b) $\hat{\theta}_k = F_k^\dagger(\hat{\sigma}_{k-1}) f_k(\hat{\sigma}_{k-1})$

c) $\hat{\sigma}_k = G^\dagger(\hat{\theta}_k) g_k(\hat{\theta}_k)$

It seems natural to initialise σ as the null vector, i.e. the algorithm starts with the (biased) LS solution of θ which is subsequently improved during the following recursion steps (provided σ is estimated appropriately). Note that Step 2b requires $O(n_z^3)$ flops whilst Step 2c is only of order $O(n_a)$.

Focus is now directed towards a recursive implementation of the overdetermined BCLS problem (5.10a) based on a generalisation of the recursive bias compensation technique (see Point 4a in Section 5.3.1).

5.3.3 Recursive bias compensating least squares approach

In order to develop a recursive algorithm based on the bilinear parametrisation, which avoids the solution of the resulting LS problems in an offline manner, the approach used in Section 3.3.2 is extended to deal with a set of overdetermined normal equations. This will reduce the computational complexity from cubic to quadratic order. The general case is summarised in the following Lemma.

Lemma 5.1 (Recursive extended bias compensation). Let Σ_k and ξ_k denote a general covariance matrix and covariance vector and let $\tilde{\Sigma}_k$ and $\tilde{\xi}_k$ denote the corresponding noise compensating terms. Given the overdetermined bias compensated normal equations

$$\left(\Sigma_k - \tilde{\Sigma}_k\right)\hat{\theta}_k = \xi_k - \tilde{\xi}_k, \tag{5.19}$$

a recursive extended bias compensating (REBC) estimator for θ_k may be expressed as

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{IV}} + \Sigma_k^{\dagger}\left(\tilde{\Sigma}_k\hat{\theta}_{k-1} - \tilde{\xi}_k\right). \tag{5.20}$$

Proof. Straightforward manipulations allow the re-arrangement of (5.19) as

$$\hat{\theta}_k = \Sigma_k^{\dagger}\left(\tilde{\Sigma}_k\hat{\theta}_k + \xi_k - \tilde{\xi}_k\right). \tag{5.21}$$

Thus, by acknowledging that $\Sigma_k^{\dagger}\xi_k$ is the (biased) IV solution $\hat{\theta}_k^{\mathrm{IV}}$, and by approximating $\hat{\theta}_k$ with $\hat{\theta}_{k-1}$, leads directly to (5.20). ∎

Consequently, by applying Lemma 5.1 to (5.10a) gives the update equation for the unbiased solution

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{IV}} + \hat{\Sigma}_{z\varphi}^{k\dagger}\left(\hat{\Sigma}_{\tilde{z}\tilde{\varphi}}^{k-1}\hat{\theta}_{k-1} - \hat{\xi}_{\tilde{z}\tilde{y}}^{k-1}\right), \tag{5.22}$$

where the compensation terms are given by

$$\Sigma_{\tilde{z}\tilde{\varphi}}(\hat{\sigma}_k) = \begin{bmatrix} 0 & 0 \\ \hat{\sigma}_{\tilde{y}}^k I_{n_a} & 0 \\ 0 & \hat{\sigma}_{\tilde{u}}^k I_{n_b} \\ 0 & 0 \end{bmatrix}, \tag{5.23a}$$

$$\xi_{\tilde{z}\tilde{y}}(\hat{\sigma}_k) = \begin{bmatrix} \hat{\sigma}_{\tilde{y}}^k \\ 0 \\ 0 \end{bmatrix}. \tag{5.23b}$$

It remains to determine the instrumental variable estimate $\hat{\theta}_k^{\mathrm{IV}}$ recursively. Therefore, the ERLS1 algorithm, as discussed in Point 4a in Section 5.3.1 is now considered.

ERLS1

The next task concerns the recursive computation of $\hat{\theta}_k^{\mathrm{IV}} = \hat{\Sigma}_{z\varphi}^{k\dagger}\hat{\xi}_{zy}^k$. Since the normal equations are overdetermined, i.e. $\hat{\Sigma}_{z\varphi}^k$ is rectangular with more rows than columns, the matrix inversion lemma (Ljung 1999, p. 364) cannot be applied directly. However, an extended version of the matrix inversion lemma for the Moore-Penrose pseudo inverse (also denoted left pseudo inverse) has been reported in (Feng et al. 2001), which allows

the formulation of a so-called extended RLS. Such an algorithm may be utilised to recursively solve an overdetermined set of normal equations.

Lemma 5.2 (Matrix pseudo inversion lemma). Define $A \in \mathbb{R}^{m\times n}$ with $\operatorname{rank}(A) = n$, $m \geq n$ as well as the vectors $b \in R^m$, $c \in R^n$. Furthermore introduce the following assumptions:

1. The matrix $A + bc^T$ is of full column rank and
2. $\mathcal{R}(A + bc^T) = \mathcal{R}(A)$,

where $\mathcal{R}(A)$ denotes the range of A. Then the left pseudo inverse of $A + bc^T$ is given by

$$[A + bc^T]^\dagger = A^\dagger - \frac{A^\dagger bc^T A^\dagger}{1 + c^T A^\dagger b}. \tag{5.24}$$

Proof. Here, only a sketch of the proof is given. For more details, the reader is referred to (Feng et al. 2001).

The post-multiplication of (5.24) with $A + bc^T$ yields the identity on both sides, which shows that the right hand side of (5.24) is *a left inverse* of $A + bc^T$. In order to show that this left inverse is the unique pseudo inverse, Assumption 2 is used. ∎

The extended recursive least squares algorithm using the matrix pseudo inversion lemma and utilising a scaled covariance matrix, such that $P_k = \hat{\Sigma}_{z\varphi}^{k\dagger}$, is given by (see Appendix G for a detailed derivation)

$$\theta_k^{\mathrm{IV}} = \theta_{k-1}^{\mathrm{IV}} + L_k \left(y_k - \varphi_k^T \theta_{k-1}^{\mathrm{IV}}\right), \tag{5.25a}$$

$$L_k = \frac{P_{k-1}\gamma_k z_k}{1 - \gamma_k + \varphi_k^T P_{k-1}\gamma_k z_k}, \tag{5.25b}$$

$$P_k = \frac{1}{1-\gamma_k}\left(P_{k-1} - L_k \varphi_k^T P_{k-1}\right). \tag{5.25c}$$

Note that the pseudo inverse in (5.22) can be replaced by P_k which yields

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{IV}} + P_k \left(\hat{\Sigma}_{\tilde{z}\tilde{\varphi}}^{k-1}\hat{\theta}_{k-1} - \hat{\xi}_{\tilde{z}\tilde{y}}^{k-1}\right). \tag{5.26}$$

The recursive bias compensating instrumental variable algorithm for overdetermined normal equations, which is denoted here as REBC, can be summarised as follows.

Algorithm 5.2 (REBC).

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{IV}} + P_k \left(\hat{\Sigma}_{\tilde{z}\tilde{\varphi}}^{k-1} \hat{\theta}_{k-1} - \hat{\xi}_{\tilde{z}\tilde{y}}^{k-1} \right) \tag{5.27a}$$

$$\hat{\theta}_k^{\mathrm{IV}} = \hat{\theta}_{k-1}^{\mathrm{IV}} + L_k \left(y_k - \varphi_k^T \hat{\theta}_{k-1}^{\mathrm{IV}} \right) \tag{5.27b}$$

$$L_k = \frac{P_{k-1}\gamma_k z_k}{1 - \gamma_k + \varphi_k^T P_{k-1}\gamma_k z_k} \tag{5.27c}$$

$$P_k = \frac{1}{1-\gamma_k} \left(P_{k-1} - L_k \varphi_k^T P_{k-1} \right) \tag{5.27d}$$

The recursive two-stage algorithm based on the bilinear parametrisation, which makes use of the REBC rule is denoted RBP2 and can be summarised as follows.

Algorithm 5.3 (RBP2).

1) Initialisation

2) For $k = n_a + n_b + 1, \ldots$

- **a)** Update $\hat{\Sigma}_{z\varphi}^k$ and $\hat{\xi}_{zy}^k$ via (5.12)
- **b)** Determine $\hat{\theta}_k$ via Algorithm 5.2
- **c)** $\hat{\sigma}_k = G_k^{\dagger}(\hat{\theta}_k) g(\hat{\theta}_k)$ using (5.13)

Remark 5.2 (Relation to recursive EBCLS developed in (Ekman 2005)). A recursive algorithm for the white noise EBCLS case has also been developed in (Ekman 2005). However, this algorithm requires the inversion of a $2n_\theta \times 2n_\theta$ matrix at each recursion step. Another conceptual difference is that the algorithm proposed in (Ekman 2005) propagates the bias compensated inverse covariance matrix (based on the current estimate $\hat{\sigma}_k$) at each time step, whereas it is the uncompensated P_k which is stored in the RBP2 algorithm. This means that the RBP2 algorithm always utilises the latest estimate of σ for the bias compensation in contrast to the algorithm in (Ekman 2005) where the errors in σ are propagated.

5.3.4 Numerical examples

The appropriateness of the developed RBP1 and the RBP2 algorithms is investigated in simulation for a particular system.

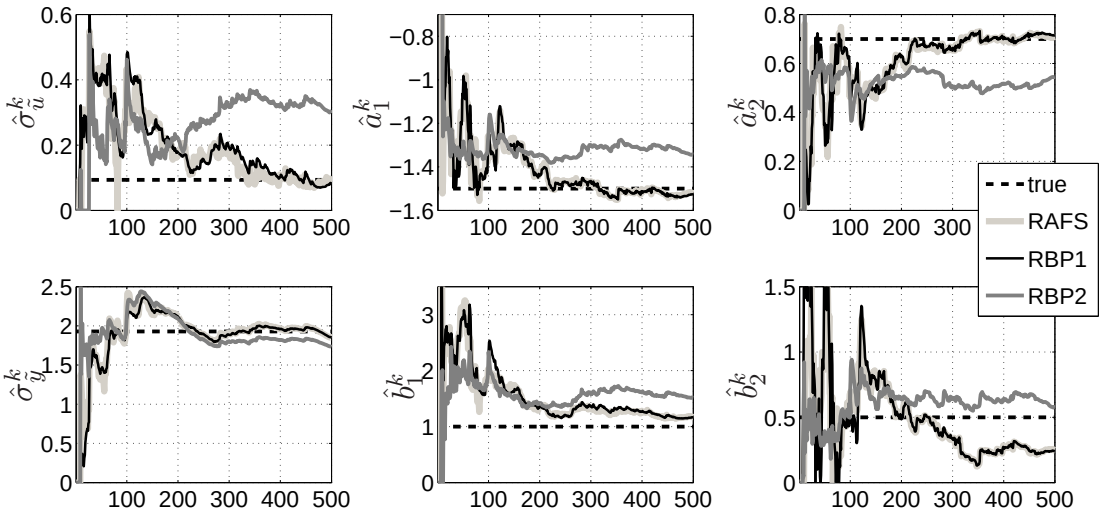


Figure 5.1: Estimates of ϑ using RBP1 and RBP2 in comparison with RAFS for Example 5.1.

Example 5.1. Consider an identical setup as in Section 3.3.6, i.e. the system given by

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix}^T, \tag{5.28a}$$

$$\sigma = \begin{bmatrix} 2.1 & 0.1 \end{bmatrix}^T, \tag{5.28b}$$

with the input being a zero mean random process of unity variance. The RBP1 and the RBP2 algorithms are utilised to estimate ϑ for $N = 500$ samples. The recursions commence at $k = n_\zeta + n_b = 8$ (cf. the definition of ζ_k in (2.50)) and the estimates of ϑ are initialised with null vectors for both algorithms. In addition, the pseudo-inverse P_k for the REBC needs to be initialised in the case of RBP2. Here, P_k when $k = 8$ is chosen to be initialised with the exact pseudoinverse of $\hat{\Sigma}^k_{z\varphi}$. The results of both recursive estimators are compared with those of the offline Frisch scheme, namely with the RAFS algorithm (see Algorithm 3.1 on page 56). Figure 5.1 shows the estimates evolving with time. It is observed that the estimates of σ obtained by the RBP1 algorithms are very similar to those obtained by the RAFS. The estimates of the parameter θ obtained by the RBP1 and RAFS are virtually identical for $k \gtrapprox 200$. The quality of the estimates obtained from the RBP2, in contrast, is rather poor. This observation is further investigated in the next example. ■

Example 5.2. The simulation of Example 5.1 is repeated for N=5000 and the RBP1 and RBP2 algorithms are applied to estimate ϑ. However, two different variants of the RBP2 algorithm are used: The first variant computes, as before, P_k using the matrix pseudo inversion lemma, whilst a second variant, denoted here RBP2$_I$ computes P_k exactly for the first 250 recursions, in order to achieve a more exact initialisation of the pseudoinverse. The mean-square-error between the RAFS estimates and those obtained

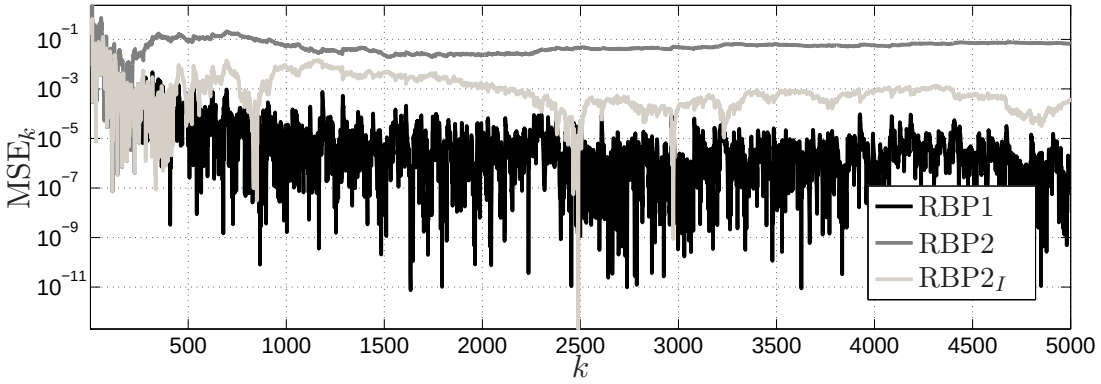


Figure 5.2: Mean square error for between RAFS and RBP1, RBP2, RBP2$_I$ evolving with time for Example 5.2.

from the RBP1, RBP2 and RBP2$_I$ are, respectively, computed as

$$\mathrm{MSE}_k \triangleq \frac{\left\| \vartheta^k_{\mathrm{RAFS}} - \hat{\vartheta}_k \right\|_2^2}{n_\theta + 2}, \tag{5.29}$$

where $\vartheta^k_{\mathrm{RAFS}}$ denotes the estimates of the RAFS. The values of MSE_k evolving over time are in shown in Figure 5.2. It is observed that the longer period of initialisation can improve the performance in terms of MSE_k for the RBP2 algorithm (from around 10^{-1} to approximately 10^{-3} with respect to the RAFS). However, the results of the RBP1 appear to be superior with an average value around 10^{-7}. ■

Examples 5.1 and 5.2 reveal an important observation which is captured in the following remark.

Remark 5.3. The REBC approach for the overdetermined normal equations (Algorithm 5.2) appears to be sensitive with respect to its initialisation of the pseudo inverse. In fact, some further numerical experiments reveal that the initialisation of P_k, the left pseudo inverse of $\hat{\Sigma}^k_{z\varphi}$, is of crucial importance in achieving a satisfactory operation of the algorithm. This requirement may be considered as a major shortcoming that has been observed for the REBC, hence the resulting RBP2 algorithm.

5.3.5 Comments on the matrix pseudo inversion lemma for recursive estimation

A potential reason for the poor performance of the ERLS1 algorithm is that the second assumption within Lemma 5.2 is not necessarily satisfied when applied to recursive estimation. In this case, the right hand side of (5.24) is a left inverse of $A + bc^T$, but not the unique pseudo inverse (cf. Chapter 7 in Lawson & Hanson 1995) as illustrated in the following example.

Example 5.3. Let A, b and c be given by

$$A = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad c = 1, \tag{5.30}$$

where the pseudo inverse of A is given by

$$A^{\dagger} = \left(\begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{5.31}$$

In addition, it holds

$$A + bc^T = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{5.32}$$

and

$$\left(A + bc^T\right)^{\dagger} = \left(\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 1 \end{bmatrix} = 0.5 \begin{bmatrix} 1 & 1 \end{bmatrix}. \tag{5.33}$$

In contrast, the right hand side of (5.24) gives

$$\begin{bmatrix} 1 & 0 \end{bmatrix} - \frac{\begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix}}{1 + \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}} = \begin{bmatrix} 1 & 0 \end{bmatrix} \neq \left(A + bc^T\right)^{\dagger}. \tag{5.34}$$

■

Hence, when Assumption 2 within Lemma 5.2 is not satisfied, the REBC algorithm does not use the unique pseudo inverse for the computation of the IV estimate $\hat{\theta}_k^{\mathrm{IV}}$. The implications for the estimator are illustrated with the following example.

Example 5.4. Consider two equations in one unknown given by

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} x + \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}, \tag{5.35}$$

where x is the scalar parameter to be estimated, whilst e_1 and e_2 are zero mean independent errors having the same variance. A left inverse of $[1\ 1]^T$ may be given by

$$H = \begin{bmatrix} \alpha & 1 - \alpha \end{bmatrix}, \tag{5.36}$$

where $\alpha \in \mathbb{R}$ can be chosen freely. This allows the formulation of an unbiased estimate as

$$\hat{x} = Hy = x + \alpha e_1 + (1 - \alpha) e_2. \tag{5.37}$$

However, due to the Gauss-Markov theorem, the best linear unbiased estimator occurs when

$$H = \begin{bmatrix} 1 \\ 1 \end{bmatrix}^{\dagger} = 0.5 \begin{bmatrix} 1 & 1 \end{bmatrix}. \tag{5.38}$$

■

Consequently, when Assumption 2 of Lemma 5.2 is not satisfied, but (5.24) is still used to update the left inverse, the estimate is no longer best in the minimum variance sense. Hence, although the ERLS1 algorithm is computationally simpler than the ERLS2 algorithm, it does not solve the same problem and leads to a degraded statistical behaviour. This implies that for the application considered here, it seems to be advantageous to make use of the ERLS2 algorithm rather than the ERLS1, in order to compute the IV estimate recursively.

5.4 Variable projection algorithm

This section considers an application of the variable projection algorithm to recursively estimate ϑ. Recent work (Ekman 2005, Söderström et al. 2005) recommends this algorithm not only for (offline) EBCLS problems, but also as an alternative for the iterative BCLS implementations as used, for instance, in the bias eliminating least squares (BELS) methods. Whilst an application of the variable projection seems to appear unnecessary for solving the Frisch-YW identification problem due to the nature of its bilinear parametrisation, it is its generalisability towards more complicated setups, such as coloured noise cases, which makes this approach appealing. In fact, only the computation of the Jacobian is to be modified, in order to apply this approach to nonlinear problems which are not bilinear in the parameters, but still separable. In particular, the approach could be utilised to obtain an alternative recursive algorithm for the Frisch scheme identification problem in the coloured output noise case, which has been considered in Chapter 4.

Since the nonlinear least squares problem (5.1) is separable (cf. Section 2.3.2), ϑ can be estimated offline in a two-step manner using a variable projection algorithm given as follows.

Algorithm 5.4 (Variable projection algorithm - offline case).

$$\hat{\sigma} = \min_{\sigma} \left\| \left(I - F(\sigma) F^{\dagger}(\sigma) \right) f(\sigma) \right\|_2^2 \tag{5.39a}$$

$$\hat{\theta} = F^{\dagger}(\hat{\sigma}) f(\hat{\sigma}) \tag{5.39b}$$

The minimisation problem in (5.39a) could be solved using any suitable optimisation

routine. However, it is beneficial to take the particular structure of the problem into account which can be realised, for instance, via a Gauss-Newton algorithm. This also has the advantage of allowing a straightforward recursive implementation. A Gauss-Newton algorithm for the variable projection problem is given in (Björck 1996) as follows:

Algorithm 5.5 (Gauss-Newton variable projection algorithm). Let $\hat{\vartheta}_k = \begin{bmatrix} \hat{\theta}_k^T & \hat{\sigma}_k^T \end{bmatrix}^T$ be the current approximation.

1. Solve the linear sub-problem

$$\hat{\theta}_{k+\frac{1}{2}} = \arg\min_{\theta_k} \ \|F(\hat{\sigma}_k)\ \theta_k - f(\hat{\sigma}_k)\|_2^2 \tag{5.40}$$

and set $\hat{\vartheta}_{k+\frac{1}{2}} = \begin{bmatrix} \hat{\theta}_{k+\frac{1}{2}}^T & \hat{\sigma}_k^T \end{bmatrix}^T$.

2. Compute the Gauss-Newton direction ψ_k at $\hat{\vartheta}_{k+\frac{1}{2}}$, i.e. solve

$$\hat{\psi}_k = \arg\min_{\psi_k} \ \left\| J(\hat{\vartheta}_{k+\frac{1}{2}})\psi_k + r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) \right\|_2^2, \tag{5.41}$$

where $J(\hat{\vartheta}_{k+\frac{1}{2}})$ is the Jacobian matrix.

3. Take

$$\hat{\vartheta}_{k+1} = \hat{\vartheta}_{k+\frac{1}{2}} - \gamma_k \hat{\psi}_k, \tag{5.42}$$

where γ_k denotes the step size.

The Jacobian matrix in (5.41) is defined by

$$J(\hat{\vartheta}_{k+\frac{1}{2}}) \triangleq \frac{\partial r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k)}{\partial \hat{\vartheta}_{k+\frac{1}{2}}} = \begin{bmatrix} \frac{\partial r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k)}{\partial \hat{\theta}_{k+\frac{1}{2}}} & \frac{\partial r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k)}{\partial \hat{\sigma}_k} \end{bmatrix}, \tag{5.43}$$

where

$$\frac{\partial r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k)}{\partial \hat{\theta}_{k+\frac{1}{2}}} = F(\hat{\sigma}_k), \tag{5.44a}$$

$$\frac{\partial r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k)}{\partial \sigma_k} = \frac{\partial F(\hat{\sigma}_k)}{\partial \sigma_k} \hat{\theta}_{k+\frac{1}{2}} - \frac{\partial f(\hat{\sigma}_k)}{\partial \hat{\sigma}_k}. \tag{5.44b}$$

Recall from (5.7) that in the Frisch-YW case, $r(\vartheta)$ is also linear in σ, i.e.

$$r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) = G(\hat{\theta}_{k+\frac{1}{2}})\hat{\sigma}_k - g(\hat{\theta}_{k+\frac{1}{2}}) \tag{5.45}$$

holds. This simplifies the computation of (5.44b) to

$$\frac{\partial r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k)}{\partial \sigma_k} = G(\hat{\theta}_{k+\frac{1}{2}}). \tag{5.46}$$

Consequently, when the problem is bilinear in θ and σ, i.e. (5.5) and (5.7) hold, the Jacobian in Step 2 simplifies to

$$J(\hat{\vartheta}_{k+\frac{1}{2}}) = \begin{bmatrix} F(\hat{\sigma}_k) & G(\hat{\theta}_{k+\frac{1}{2}}) \end{bmatrix}. \tag{5.47}$$

As in Section 5.3, the problem reduces to solving two individual linear least squares problems, which can be solved in different ways as pointed out in Section 5.3.1. Two approaches are considered in this Section: The first proposed algorithm uses robust batch LS techniques and is therefore of cubic complexity, i.e. $O((n_\theta + 2)^3)$. The algorithm is denoted RVP1 (recursive variable projection) and summarised as follows:

Algorithm 5.6 (RVP1).

1) Set $j = n_\zeta + n_b$, $\hat{\sigma}_j = 0$

2) For $k = j + 1, ...$

- **a)** Update $\hat{\Sigma}^k_{z\varphi}$ and $\hat{\xi}^k_{zy}$ via (5.12)
- **b)** $\hat{\theta}_{k+\frac{1}{2}} = F^\dagger(\hat{\sigma}_k) f(\hat{\sigma}_k)$
- **c)** $\hat{\psi}_k = J^\dagger(\hat{\vartheta}_{k+\frac{1}{2}}) r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k)$
- **d)** $\hat{\vartheta}_{k+1} = \hat{\vartheta}_{k+\frac{1}{2}} - \gamma_k \hat{\psi}_k$

In order to reduce the computational complexity of the RVP1 algorithm, the recursive implementation of the two linear LS problems based on a generalisation of the recursive bias compensation technique and the matrix inversion lemma for pseudo inverses, as in Section 5.3, is considered, leading to the second approach.

5.4.1 Update of $\hat{\theta}_{k+\frac{1}{2}}$

Step 1 of Algorithm 5.5 can be solved via an extended bias-compensating linear least squares approach (cf. Lemma 5.1 on page 128) for which a recursive algorithm has been derived in Section 5.3 (Algorithm 5.2). The algorithm for the determination of $\hat{\theta}_{k+\frac{1}{2}}$ can be summarised as follows.

Algorithm 5.7 (Update of $\hat{\boldsymbol{\theta}}_{\boldsymbol{k+\frac{1}{2}}}$).

$$\hat{\theta}_{k+\frac{1}{2}} = \hat{\theta}^{\mathrm{IV}}_{k+\frac{1}{2}} + P_k \left(\Sigma_{\tilde{z}\tilde{\varphi}} \hat{\theta}_{k-\frac{1}{2}} - \xi_{\tilde{z}\tilde{y}} \right) \tag{5.48a}$$

$$\hat{\theta}^{\mathrm{IV}}_{k+\frac{1}{2}} = \hat{\theta}^{\mathrm{IV}}_{k-\frac{1}{2}} + L_k \left(y_k - \varphi_k^T \hat{\theta}^{\mathrm{IV}}_{k-\frac{1}{2}} \right) \tag{5.48b}$$

$$L_k = \frac{P_{k-1}\gamma_k z_k}{1 - \gamma_k + \varphi_k^T P_{k-1} \gamma_k z_k} \tag{5.48c}$$

$$P_k = \frac{1}{1-\gamma_k} \left(P_{k-1} - L_k \varphi_k^T P_{k-1} \right) \tag{5.48d}$$

Next, the update of the Gauss-Newton update direction is considered.

5.4.2 Update of $\hat{\psi}_k$

Step 2 in Algorithm 5.5 can be expressed as

$$\begin{bmatrix} F(\hat{\sigma}_k) & G(\hat{\theta}_{k+\frac{1}{2}}) \end{bmatrix} \begin{bmatrix} \psi^\theta \\ \psi^\sigma \end{bmatrix} = r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k), \tag{5.49}$$

where ψ^θ and ψ^σ denote the Gauss-Newton directions corresponding to θ and σ, respectively. A recursive solution of the linear LS problem (5.49) is, however, not straightforward: Since the matrix $[F(\hat{\sigma}_k)\ G(\hat{\theta}_{k+1/2})]$ is not updated via a common rank-1 update, the matrix (pseudo) inversion lemma is not applicable. To overcome this issue, an iterative two-step solution is proposed which first computes the solution for ψ^θ followed by a solution for ψ^σ, i.e.

$$\hat{\psi}^\theta_k = F^\dagger(\hat{\sigma}_k) \left[r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) - G(\hat{\theta}_{k+\frac{1}{2}}) \hat{\psi}^\sigma_{k-1} \right], \tag{5.50a}$$

$$\hat{\psi}^\sigma_k = G^\dagger(\hat{\theta}_{k+\frac{1}{2}}) \left[r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) - F(\hat{\sigma}_k) \hat{\psi}^\theta_k \right]. \tag{5.50b}$$

For the determination of $\hat{\psi}^\theta_k$ the principle of stationary iterative LS methods (cf. Section 2.3.1) can be applied. From (5.50a), one obtains,

$$\left[\hat{\Sigma}^k_{z\varphi} - \Sigma_{\tilde{z}\tilde{\varphi}}(\hat{\sigma}_k) \right] \hat{\psi}^\theta_k = r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) - G(\hat{\theta}_{k+\frac{1}{2}}) \hat{\psi}^\sigma_{k-1}, \tag{5.51}$$

which gives rise to the recursive update

$$\hat{\psi}^\theta_k = P_k \left[r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) - G(\hat{\theta}_{k+\frac{1}{2}}) \hat{\psi}^\sigma_{k-1} + \Sigma_{\tilde{z}\tilde{\varphi}}(\hat{\sigma}_k) \psi^\theta_{k-1} \right], \tag{5.52}$$

where P_k has already been computed in Algorithm 5.7. Due to the particular sparse structure of $G(\hat{\theta}_{k+\frac{1}{2}})$, its pseudo inverse and hence $\hat{\psi}^\sigma_k$ in (5.50b) can be computed in

a straightforward manner as given in (5.13)

$$G^{\dagger}(\hat{\theta}_{k+\frac{1}{2}}) = \begin{bmatrix} \frac{1}{1+\hat{a}^T_{k+\frac{1}{2}}\hat{a}_{k+\frac{1}{2}}} & \frac{-\hat{a}^T_{k+\frac{1}{2}}}{1+\hat{a}^T_{k+\frac{1}{2}}\hat{a}_k} & 0 & 0 \\ 0 & 0 & \frac{-\hat{b}^T_{k+\frac{1}{2}}}{\hat{b}^T_{k+\frac{1}{2}}\hat{b}_{k+\frac{1}{2}}} & 0 \end{bmatrix}. \tag{5.53}$$

The resulting algorithm is summarised as follows.

Algorithm 5.8 (Update of $\hat{\psi}_k$).

$$\hat{\psi}_k = \begin{bmatrix} \psi^{\theta T} & \psi^{\sigma T} \end{bmatrix}^T \tag{5.54a}$$

$$\hat{\psi}^{\theta}_k = P_k \left[r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) - G(\hat{\theta}_{k+\frac{1}{2}})\hat{\psi}^{\sigma}_{k-1} + \Sigma_{\tilde{z}\tilde{\varphi}}(\hat{\sigma}_k)\psi^{\theta}_{k-1} \right] \tag{5.54b}$$

$$\hat{\psi}^{\sigma}_k = G^{\dagger}(\hat{\theta}_{k+\frac{1}{2}}) \left[r(\hat{\theta}_{k+\frac{1}{2}}, \hat{\sigma}_k) - F(\hat{\sigma}_k)\hat{\psi}^{\theta}_k \right] \tag{5.54c}$$

5.4.3 Algorithm summary

The recursive algorithm, denoted RVP2, can be summarised as follows.

Algorithm 5.9 (RVP2).

1) Initialisation

2) For $k = n_a + n_b + 1, ...$

- **a)** Update $\hat{\Sigma}^k_{z\varphi}$ and $\hat{\xi}^k_{zy}$ via (5.12)
- **b)** Determine $\hat{\theta}_{k+\frac{1}{2}}$ via Algorithm 5.7
- **c)** Compute $\hat{\psi}_k$ using Algorithm 5.8
- **d)** Update $\hat{\vartheta}_{k+1} = \hat{\vartheta}_{k+\frac{1}{2}} - \gamma_k \hat{\psi}_k$

Note that in contrast to the RVP1, Algorithm 5.9 is of only quadratic complexity.

The following example investigates, if the RVP2 algorithm can approximate the estimates of the more computationally demanding RVP1 algorithm.

Example 5.5. Consider an identical setup as in Section 3.3.6, i.e. the system is given

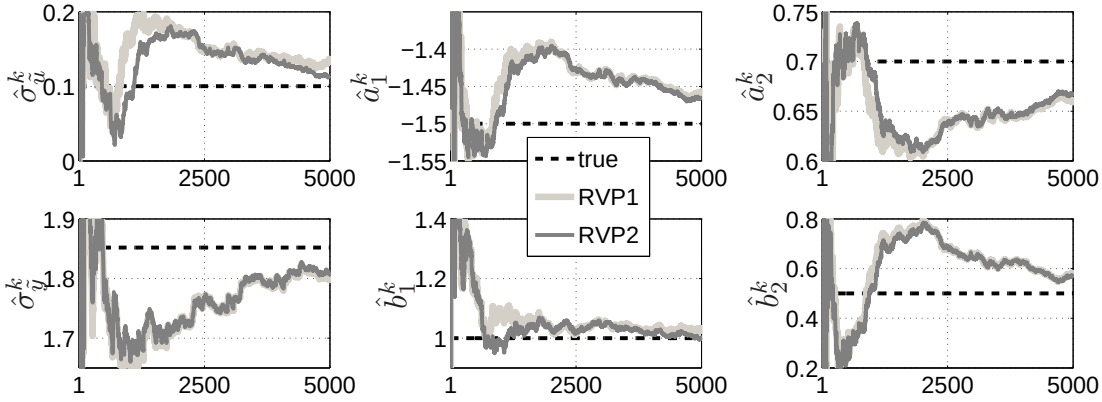


Figure 5.3: Estimates of ϑ using RVP1 and RVP2 for Example 5.5.

by

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix}^T, \tag{5.55a}$$

$$\sigma = \begin{bmatrix} 2.1 & 0.1 \end{bmatrix}^T, \tag{5.55b}$$

with the input being a zero mean random process of unity variance. The signal-to-noise ratio on input and output is given by 10dB and 10dB, respectively. The RVP1 and the RVP2 algorithms are utilised to estimate ϑ. During the first 100 recursions, P_k within the RVP2 algorithm is computed exactly, rather than by making use of the matrix pseudo inversion lemma. The estimates of both algorithms evolving over time are shown in Figure 5.3 for $N = 5000$ samples. It is observed that the RVP2 algorithm can successfully approximate the estimates obtained by the RVP1 algorithm. However, the initialisation of P_k is, as already remarked in Section 5.3.4, crucial for the RVP2 algorithm to perform satisfactorily. ■

In the next section the four EBCLS-based algorithms which have been developed in this chapter are compared with those developed in earlier chapters in terms of robustness, accuracy, Frisch-character as well as computation time.

5.5 Simulation studies

This section compares the recursive EBCLS algorithms of this chapter with the RFS and FRFS approaches, which have been developed in Chapters 3 and 4, respectively. For convenience, a brief description of all eight algorithms, which are compared in this section, are listed in Table 5.1 together with the appropriate sections where the details can be found. Three simulation examples are designed such that the following aspects are investigated:

Computation time: For an online implementation it is necessary to gather information about the computation time per recursion, consequently this aspect is

Algorithm	Description	Section
RFSa RFSb	Gradient-based recursive Frisch-YW.	3.3
FRFSa FRFSb	Fast RFS algorithms.	4.2
RBP1 RBP2	Recursive bilinear parametrisation.	5.3
RVP1 RVP2	Recursive variable projection.	5.4

Table 5.1: Algorithms used within simulation.

investigated for an incrementally increasing model order.

Frisch-character & estimation accuracy: Whilst the RFS algorithms aim to determine $\sigma_{\tilde{y}}$ such that $\Sigma_{\bar{\varphi}_0}$ is (approximately) singular, the EBCLS approaches do not explicitly impose such a condition on the solution. It is therefore of interest to investigate how 'well' the EBCLS based recursive algorithms satisfy this singularity condition. To evaluate this property, a Monte-Carlo simulation is performed, which, in addition, monitors the estimation accuracy of the various algorithms. Note that the asymptotic accuracy of all eight algorithms is expected to be similar[4] (convergence provided) since the same underlying equations are utilised (Hong et al. 2007, Hong & Söderström 2008).

Robustness & convergence: A simulation example is included to give some insight into the robustness and convergence behaviour for a particular system. In an attempt to quantify the results, the MSEs of the estimates are monitored in a Monte-Carlo simulation for an increasing signal-to-noise ratios (SNRs) on input and output, respectively.

Example 5.6 (Computation time). A similar setup as in Section 3.4.3 (page 77) and Example 4.3 (page 104) is considered, where the computation times of the RAFS, RFS and FRFS algorithms have been compared. For the considered algorithms, the computation time per recursion for model orders ranging from $n = 1, ..., 30$ is presented in Figure 5.4. The solid lines are utilised for algorithms which are of cubic complexity, whilst the fast versions, which are of quadratic complexity, have dashed lines. It is observed that the RFS algorithms are the most expensive algorithms in terms of computation time per recursion. The slopes of the curves corresponding to the RBP1 and RVP1 indicates that these algorithms are of cubic complexity whereas the lowers slope of the RBP2 and RVP2 approaches indicate the quadratic order of complexity. This confirms the theoretical results obtained in this chapter (see Sections 5.3 and 5.4). However, although of cubic complexity, the RBP1 is relatively fast and outperforms

[4]Note that the weighting is different.

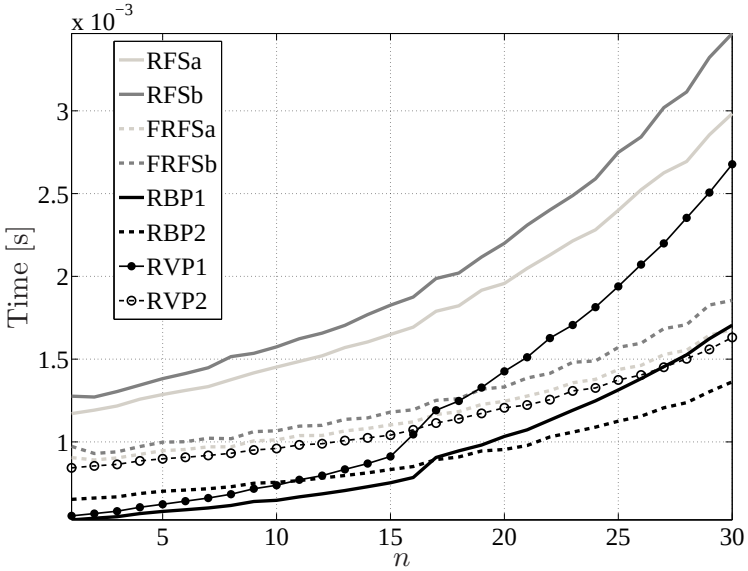


Figure 5.4: Computation time per single recursion with increasing model order n.

the other algorithms with respect to computing time for a model order less than approximately 17. For a model order greater than this threshold, the RBP2 algorithm is the fastest. A similar behaviour is observed when comparing the RVP1 and RVP2 algorithms. For a low model order, the RVP1 is faster, whilst for $n \gtrapprox 17$, the RVP2 requires less computation time. It is interesting to observe, that for the considered range of model order, the RBP1, RBP2 and the RVP2 are all less computationally demanding than the FRFS algorithms. ■

Example 5.7 (Frisch-character and estimation accuracy). Consider an identical setup as in Section 3.3.6, i.e. the system given by

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix}^T, \tag{5.56a}$$

$$\sigma = \begin{bmatrix} 2.1 & 0.1 \end{bmatrix}^T, \tag{5.56b}$$

with the input being a zero mean random process of unity variance. The system is simulated for $N = 1000$ samples and 100 Monte-Carlo iterations. The estimates of $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$ are projected into the intervals $[0, \sigma_{\tilde{u}}^{\max}]$ and $[0, \sigma_{\tilde{y}}^{\max}]$, respectively. The maximal admissible values for the input and output measurement noise variances are chosen to be $\sigma_{\tilde{u}}^{\max} = 2\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}^{\max} = 2\sigma_{\tilde{y}}$. In addition, recall from Examples 5.1-5.5 that the RBP2 and the RVP2 algorithms require an accurate initialisation of the pseudo inverse P_k. Therefore, these algorithms use the exact computation of P_k during the first 250 recursions (rather than making use of the matrix pseudo inversion lemma). The Frisch-character is measured via F_2^k as defined by (3.58), i.e. the smallest eigenvalue of $\hat{\Sigma}_{\varphi_0}^k = \hat{\Sigma}_{\varphi}^k - \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k)$. The average of F_2^k for the last 100 samples, i.e. $901 \leq k \leq N$,

Algorithm	mean($\bar{F}$)	std($\bar{F}$)	mean(MSE$_N$)	std(MSE$_N$)
RFSa	$4.41{\cdot}10^{-6}$	$1.65{\cdot}10^{-6}$	$7.19{\cdot}10^{-3}$	$9.61{\cdot}10^{-3}$
RFSb	$8.36{\cdot}10^{-6}$	$3.51{\cdot}10^{-6}$	$7.10{\cdot}10^{-3}$	$9.77{\cdot}10^{-3}$
FRFSa	$3.21{\cdot}10^{-3}$	$6.85{\cdot}10^{-4}$	$7.86{\cdot}10^{-3}$	$1.04{\cdot}10^{-2}$
FRFSb	$2.93{\cdot}10^{-3}$	$7.24{\cdot}10^{-4}$	$8.46{\cdot}10^{-3}$	$1.01{\cdot}10^{-2}$
RBP1	$1.61{\cdot}10^{-4}$	$5.65{\cdot}10^{-4}$	$7.43{\cdot}10^{-3}$	$1.00{\cdot}10^{-2}$
RBP2	$5.05{\cdot}10^{-4}$	$1.38{\cdot}10^{-3}$	$8.62{\cdot}10^{-3}$	$1.19{\cdot}10^{-2}$
RVP1	$8.02{\cdot}10^{-4}$	$3.90{\cdot}10^{-3}$	$7.58{\cdot}10^{-3}$	$1.04{\cdot}10^{-2}$
RVP2	$7.95{\cdot}10^{-3}$	$4.54{\cdot}10^{-2}$	$1.04{\cdot}10^{-2}$	$1.41{\cdot}10^{-2}$

Table 5.2: Mean (mean(·)) and standard deviation (std(·)) of $\bar{F}$ and MSE$_N$ for 100 Monte-Carlo runs.

given by

$$\bar{F} \triangleq \frac{1}{100} \sum_{k=901}^{1000} F_2^k \tag{5.57}$$

is computed for each algorithm and each Monte-Carlo run. In order to evaluate the estimation accuracy of the individual algorithms, the accuracy of the final estimate $\hat{\theta}_N$ is measured via the mean square error given by

$$\mathrm{MSE}_N = \frac{\left\| \vartheta - \hat{\vartheta}_N \right\|_2^2}{n_\theta + 2} \tag{5.58}$$

for each Monte-Carlo run. The means and standard deviations of $\bar{F}$ and MSE$_N$ with respect to the Monte-Carlo simulations for each algorithm are presented in Table 5.2. Consideration is first given to the Frisch-character, as indicated by the measure $\bar{F}$. It is observed, that the RFS algorithms, which both use the conjugate gradient subspace tracking algorithm for the determination of $\sigma_{\tilde{y}}$, yield the best performance in terms of Frisch-character. Comparing the results of these two algorithms, it is noteworthy that the Frisch-character of the RFSb is worse (approximately double) than that of the RFSa algorithm. At the first glance, this difference might appear surprising, since both algorithms utilise the CG-RQ algorithm (cf. Algorithm 3.4 on page 61) to estimate $\sigma_{\tilde{y}}$, which finally defines the Frisch-character. However, recall that the CG-RQ assumes that the matrix $\hat{A}_k$, hence $\hat{\sigma}_{\tilde{y}}^k$, is varying slowly with time, as stated in Assumption **AE1** (cf. Section 3.3.3). In addition, it has been observed in previous examples (cf. Section 3.3.6), that the estimate of $\sigma_{\tilde{u}}$ obtained from the RFSb (cf. Algorithm 3.6 on page 68) appears to more erratic than that obtained from the RFSa. Consequently, Assumption **AE1** is 'less satisfied' in the RFSb case, which could explain the deterioration of Frisch-

character which is observed in Table 5.2. The RBP approaches perform second best in terms of Frisch-character, but exhibit a relatively high standard deviation, due to several outliers[5]. The performances of the RVP1 algorithm would appear to be next following the RBP, with the FRFS algorithms providing very similar jointly next best results. It is noted that the RVP2 algorithm performs worst in terms of Frisch character.

For the estimation accuracy, it is observed that the values of the MSE_N are similar in all cases. This is an expected result, since all algorithms use the same underlying equations. ■

Example 5.8 (Robustness & convergence). In order to compare the convergence properties of the eight algorithms, the system given in Example 5.1, where ϑ was given by

$$\theta = \begin{bmatrix} -1.5 & 0.7 & 1 & 0.5 \end{bmatrix}^T, \tag{5.59}$$

is simulated for different SNRs. The SNR on the input and output, which are denoted SNRu and SNRy, respectively, are varied incrementally between 0dB and 20dB. For each combination of SNRu and SNRy, 100 Monte-Carlo iterations are performed using $N = 500$ samples. Otherwise, an identical setup as in Example 5.1 is used, in particular the pseudoinverse within the RBP2 and RVP2 approaches is computed exactly during the first 250 recursions. At each Monte-Carlo iteration, the estimation accuracy of each algorithm measured by the MSE_N defined in (5.58) is recorded. The mean value with respect to the Monte-Carlo iterations of this quantity indicates, whether the given algorithm is able to converge or not: A large value will be interpreted as divergence, whilst a small mean value will be interpreted as convergence. In order to visualise the results, a threshold is defined which declares the certain divergence of the algorithm (values above this threshold are clipped). The value of this threshold is chosen to be $1/(n_\theta)||\theta||_2^2 \approx 0.12$ (cf. (5.59)). The mean values of MSE_N for all eight algorithms are shown in Figure 5.5, where a light colour indicates a small average MSE_N for a given set of SNRu and SNRy, whilst a dark colour corresponds to a large average estimation error (black corresponds to the threshold value 0.12).

It is observed that the two gradient-based Frisch scheme approaches (RFSa and RFSb), which have been developed in Chapter 3, exhibit a similar region of convergence for the particular system considered, indicated by the light area. Furthermore, it is observed that the additional assumptions and approximations, which have been used for the development of the fast algorithms FRFSa and FRFSb, lead to a reduced convergence region. Again, the FRFSa and FRFSb yield a very similar performance. An interesting fact is observed when comparing the RBP1 and RBP2: In terms of convergence region, the RBP2 seems to be superior to the RBP1. This is a surprising result, since the RBP2 aims to compute the RBP1 estimate in a fast manner, by

[5]Note that the outliers are in terms of Frisch-character and not in terms of estimation accuracy.

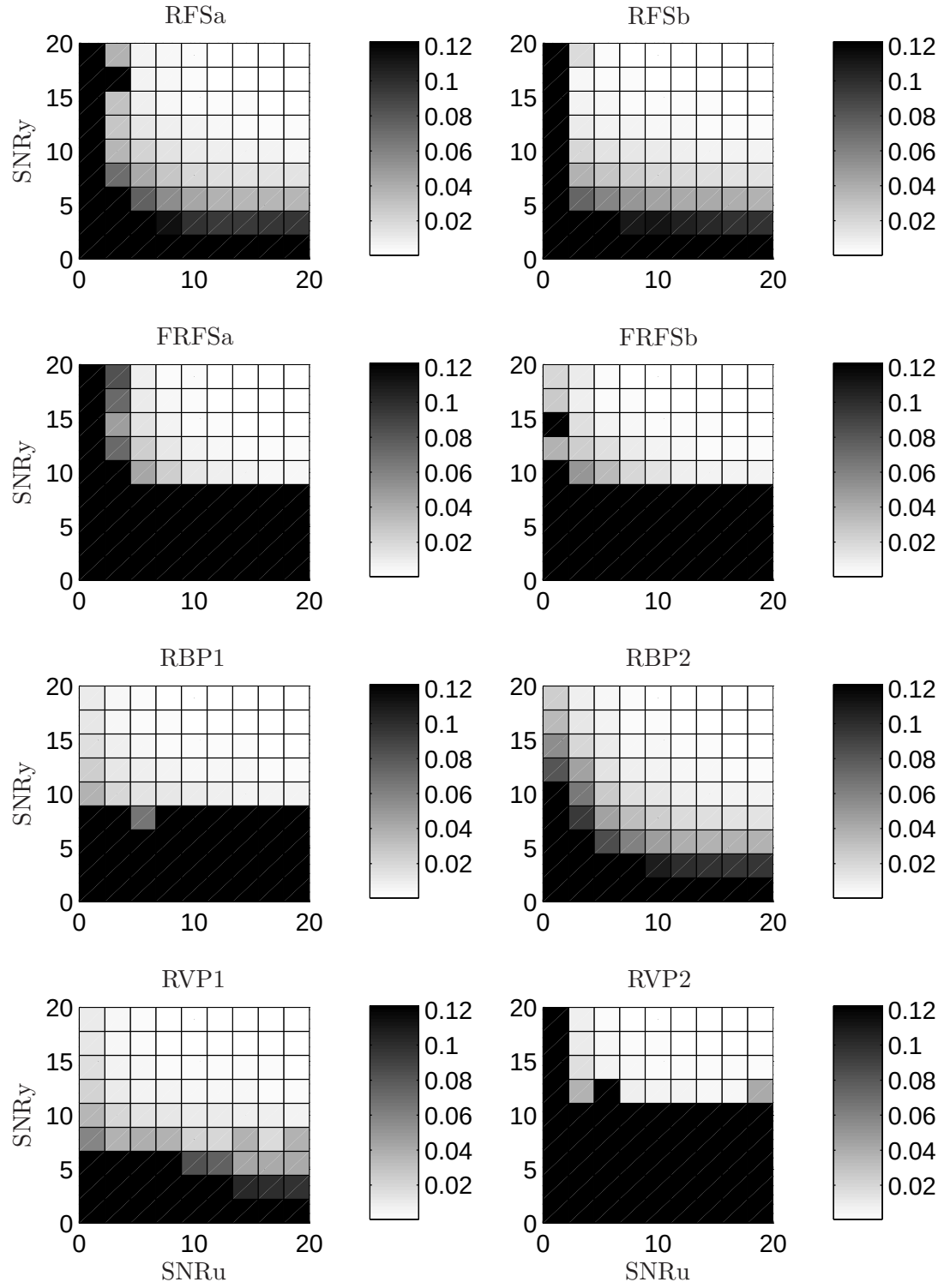


Figure 5.5: Mean values of MSE_N with respect to 100 Monte-Carlo iterations for all eight algorithms considered in Example 5.8 indicating the region of convergence. A light colour denotes a small average MSE_N, whilst a dark colour corresponds to a large mean estimation error.

introducing additional approximations. Whilst this potentially requires some further investigations, it is believed that this result may be caused by the rather more favourable initialisation of the RBP2 scheme. In addition, it is observed that the RVP2 clearly has a reduced convergence region compared to the RVP1. Comparing all investigated algorithms, it can be concluded that the RVP1 appears to yield the best convergence properties, followed by the RFS algorithms. Finally, it is worth mentioning that these simulations results cannot be interpreted as a convergence analysis, but are rather of an indicative character. Techniques for a more rigourous mathematical analysis are discussed in (Ljung & Söderström 1983, Kushner & Yin 2003) and are identified as potential further work. ■

5.6 Concluding remarks

The equations of the Frisch scheme using the Yule-Walker (YW) model selection criterion (Frisch-YW) have been interpreted in an extended bias compensating least squares (EBCLS) framework. Two methods have been considered to solve the resulting EBCLS system identification problem in a recursive manner:

1. Bilinear parametrisation approach.

2. Nonlinear separable least squares (also known as the variable projection method).

Acknowledging the fact that the EBCLS problem for the Frisch-YW case is bilinear in the parameters allows the solution of the identification problem to be obtained in an iterative two-step manner, which can easily be modified towards a recursive estimator. At each recursion, two individual least squares (LS) problems are required to be solved. A first algorithm, denoted RBP1 (recursive bilinear parametrisation), solves these LS problems in an offline or batch manner. A second recursive algorithm based on the bilinear parametrisation approach has been developed, which is denoted RBP2. This algorithm aims to avoid the application of a batch LS at each recursion by means of a recursive bias compensating least squares (RBCLS) approach; a technique which is well known in the literature and which has also been used within the previously developed RFS and FRFS algorithms. The use of the RBCLS is able to reduce the cubic complexity of the RBP1 (which is due to the batch LS application at each iteration) towards quadratic order. In order to achieve this, the RBCLS technique has been extended to deal with overdetermined normal equations by making use of the extended RLS (ERLS) which relies on the matrix pseudo inversion lemma. However, the RBCLS techniques can suffer from convergence problems, hence reducing the convergence properties of the overall RBP2 algorithm. In addition, it appears that the Moore-Penrose pseudo inverse, which is propagated within the ERLS, requires a highly accurate initialisation, in order to ensure that the RBP2 and RVP2 perform satisfactorily. Consequently, whilst providing reduced computational complexity, these issues could well represent major

limitations of the RBP2 approach. In addition, a simulation example indicates that the RBP2 fails to reduce the computation time in comparison to the RBP1 for low model orders. Hence, the practical use of the RBP2 appears to be limited.

In addition to the bilinear parametrisation approaches, two algorithms have been developed based on the variable projection method. Whilst the Frisch-YW problem is a special case within the EBCLS framework, which can be solved by the conceptually simpler bilinear parametrisation, using a variable projection allows more general setups, such as coloured noise sequences. The developed algorithms within this framework are based on a Gauss-Newton search to minimise the associated variable projection cost function. This allows a straightforward modification for a recursive application by updating the covariance matrices/vectors as new data arrives. As in the bilinear parametrisation case, the problem reduces to two separate LS problems and as in the RBP case, two algorithms have been proposed. The first algorithm is denoted RVP1 (recursive variable projection) and can be considered as being analogous to the RBP1, since the LS problems are solved in an offline manner at each recursion. The second algorithm by analogy, which corresponds to the RBP2, has been similarly denoted RVP2. It also makes use of the RBCLS and, therefore, the same implications and restrictions in terms of convergence and robustness apply.

The four algorithms which have been developed in this Chapter have been compared with the previously derived RFS and FRFS algorithms (cf. Chapters 3 and 4) via Monte-Carlo simulations. The computation time, the Frisch-character, estimation accuracy as well as an indication of convergence behaviour has been investigated. Firstly, the absolute computation time per recursion has been compared for an increasing model order. It has been found that the fast algorithms RBP2 and RVP2 are only superior in terms of computation time for an increasing model order, whilst the LS based approaches RBP1 and RVP1 are computationally less expensive for low model orders. Within the overall comparison, however, the fast RBP2 and RVP2 approaches are consistently faster than the FRFS approaches. For the Frisch-character, it has been found that the EBCLS approaches produce performances that are similar to the FRFS algorithms, whilst the RFS algorithms perform best. Finally, the region of convergence has been determined for a given system. Here, it has been found that the RVP1 yields the best performance followed by the RFS approaches. All algorithms, however, can fail to converge in the case of extremely low signal-to-noise ratios. Whilst these simulation results are required to be confirmed by mathematically sound convergence analyses, the development of recursive algorithms with improved convergence properties is an area of potential further work. In general, it can be concluded that the algorithms appear to be characterised by a trade-off between computational complexity and satisfaction of the Frisch-character. Within an overall comparison, albeit limited to a single system setup, the RVP1 seems to provide an appropriate compromise between computation time, Frisch-character and convergence properties.

Chapter 6

Errors-in-variables Kalman filtering for bilinear systems

Contents

Nomenclature

e_k	Reformulated output noise
$\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, $\mathcal{G}$, $\mathcal{N}$	System matrices of bilinear time-invariant state space representation
$\mathcal{A}_k^a$	Actual input dependent system matrix
$\mathcal{A}_k^d$	Design value for system matrix
$\hat{\mathcal{A}}_k$	Estimate of system matrix
K_k	Kalman gain
M_u, M_y	Filter performance indices for input and output, respectively
P_0	Covariance matrix of initial state vector

$P_{k\|k-1}$	State estimation error covariance matrix
P_u^k	Estimation error variance of the input
P_y^k	Estimation error variance of the output
v_k, $\bar{v}_k$, v_k^*, v_k'	Reformulated process noise
w_k	Process noise
x_k	System state vector
$\bar{x}_0$	Mean of initial state vector
$\tilde{x}_k$	State estimation error
$\hat{x}_{k\|k-1}$	One-step-ahead prediction of the state vector
z_k	Reformulated system output
ε_k	Innovation
$\sigma_{\tilde{u}}$, $\sigma_{\tilde{y}}$, $\sigma_{\tilde{u}\tilde{y}}$	Variances and covariance of input and output noise sequences
σ_e	Variance of reformulated output noise
Σ_ε^k	Innovation covariance matrix
σ_w	Variance of process noise
σ_{u_0}, σ_{y_0}	Variance of noise-free input and noise-free output, respectively
σ_u, σ_y	Variance of measured input and measured output, respectively
Σ_v^k, $\Sigma_{\bar{v}}^k$, $\Sigma_{v^*}^k$, $\Sigma_{v'}^k$	Auto-covariance matrix of reformulated process noise
Σ_{ve}^k, $\Sigma_{\bar{v}e}^k$, $\Sigma_{v^*e}^k$, $\Sigma_{v'e}^k$ ...	Cross-covariance matrix of reformulated process noise and reformulated output noise
Σ_x^k	Auto-covariance matrix of the state vector
$\Sigma_{x\tilde{x}}^k$	Cross-covariance matrix of the state vector and state estimation error
$\rho(A)$	Spectral radius of matrix A

Preliminary reading: Sections 2.2, 2.5.1, 2.5.2, 2.5.3.

6.1 Introduction

The estimation of signals from noise corrupted measurements, usually termed filtering, is a very active research area and modern references date back to the 1940s (see Ch. 1 in (Anderson & Moore 1979) for a historical development of filtering theory). The estimation of unobserved states of a linear dynamic state space system has numerous applications in automatic control, signal processing and many other areas. In the linear Gaussian case, the optimal solution, in the minimum variance sense, is given by the so-called Kalman filter (KF), which has been developed in (Kalman 1960). Recently, Kalman filtering within an errors-in-variables (EIV) framework, that is not only the output signals but also the inputs of a dynamical system are corrupted by additive measurement noise, has been considered (Guidorzi et al. 2003, Diversi et al. 2005, Markovsky & De Moor 2005). Essentially, EIV Kalman filtering provides, in addition to the state and output estimates, an estimate of the noise-free system input if the latter is unknown and only a noise corrupted measurement is available. Hence the errors-in-variables Kalman filter (EIVKF) can be considered as a generalisation of the KF which allows for a symmetric system description.

Whilst EIV filtering has only been considered for linear systems (cf. Section 2.5.2) within the literature, this chapter addresses the EIV filtering problem for bilinear sys-

tems, an appealing class of nonlinear systems. Bilinear system representations (cf. Section 2.5.3) have successfully been applied in numerous areas such as engineering, socioeconomics, chemistry and biology (Mohler 1991) since they are able to approximate many dynamical processes. Hence, there is a natural interest and motivation to extend the linear EIV filtering theory towards the bilinear case.

This chapter considers EIV Kalman filtering, i.e. the estimation of noise-free inputs and outputs when both quantities are subject to measurement noise, for a class of bilinear discrete-time dynamical single-input single-output (SISO) systems. The SISO restriction is adopted for convenience only, and the development within this chapter can be extended to the multivariate case in a straightforward manner (cf. Remark 2.4). Four suboptimal filtering approaches are considered and compared with a benchmark filter whose design depends on the knowledge of the true input, which as a consequence is an infeasible solution in practice. In addition, a connection between bilinear EIV system representations and linear time-varying systems with stochastic parameters (also known as multiplicative state noise systems) is established. A numerical example compares the performance of the developed filters. Part of the material within this chapter has been published in (Linden, Vinsonneau & Burnham 2007*a*).

6.2 Preliminaries

Assume that the input-output data is generated by a SISO bilinear EIV state space system which is given, for $k \geq 0$, by

$$x_{k+1} = \mathcal{A}x_k + \mathcal{B}u_{0_k} + \mathcal{N}u_{0_k}x_k + \mathcal{G}w_k, \qquad x_0 = \bar{x}_0, \tag{6.1a}$$

$$y_{0_k} = \mathcal{C}x_k + \mathcal{D}u_{0_k}, \tag{6.1b}$$

$$u_k = u_{0_k} + \tilde{u}_k, \tag{6.1c}$$

$$y_k = y_{0_k} + \tilde{y}_k, \tag{6.1d}$$

where $x_k \in \mathbb{R}^{n_x}$ denotes the state vector and the system matrices $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, $\mathcal{G}$ and $\mathcal{N}$, are known, constant quantities of appropriate dimension, whilst u_k and y_k are assumed to be scalars. The following assumptions are introduced.

AN7 The noise sequences $\tilde{u}_k$, $\tilde{y}_k$ and w_k are assumed to be zero mean, white, independent of u_{0_k} and are characterised by the known covariance matrices

$$E\left[\begin{bmatrix} x_0 \\ \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix}\begin{bmatrix} x_0^T & \tilde{u}_l & \tilde{y}_l & w_l \end{bmatrix}\right] = \begin{bmatrix} P_0 & 0 & 0 & 0 \\ 0 & \sigma_{\tilde{u}} & \sigma_{\tilde{u}\tilde{y}} & 0 \\ 0 & \sigma_{\tilde{u}\tilde{y}} & \sigma_{\tilde{y}} & 0 \\ 0 & 0 & 0 & \sigma_w \end{bmatrix}\delta_{kl}. \tag{6.2}$$

The problem considered in this chapter can be summarised as follows.

Problem 6.1 (EIV filtering for bilinear systems). Given the bilinear EIV system (6.1), the covariance matrices defined in **AN7** and an increasing sequence of measured pairs of inputs and outputs $\{u_i, y_i\}_{i=1}^{k}$, determine, at each time step k, the estimates of the noise-free inputs and outputs, denoted $\hat{u}_{0_k}$ and $\hat{y}_{0_k}$, respectively.

In order to proceed in a similar manner to the linear case, the system representation (6.1) is transformed into

$$x_{k+1} = \mathcal{A}x_k + \mathcal{B}u_k + \mathcal{N}u_k x_k - \mathcal{B}\tilde{u}_k - \mathcal{N}\tilde{u}_k x_k + \mathcal{G}w_k, \qquad x_0 = \bar{x}_0, \tag{6.3a}$$

$$z_k = \mathcal{C}x_k + e_k, \tag{6.3b}$$

where

$$z_k = y_k - \mathcal{D}u_k, \tag{6.4a}$$

$$e_k = \tilde{y}_k - \mathcal{D}\tilde{u}_k. \tag{6.4b}$$

A natural approach would be to follow a similar strategy as in the non-EIV bilinear case and to recast the problem, such that the standard KF can be applied (cf. Algorithm 2.2 on page 39). However, the problem is exacerbated by the additional term $-\mathcal{N}\tilde{u}_k x_k$ in (6.3a) which can be dealt in two different ways: the system can be interpreted as having an uncertain system matrix or a state dependent noise term. This is discussed in the following subsections.

6.2.1 Linear time varying system with uncertain system matrix

The first possibility is to interpret the bilinear EIV system (6.3) as a system with time-varying but uncertain system matrix, i.e.

$$x_{k+1} = \mathcal{A}_k^a x_k + \mathcal{B}u_k + v_k, \tag{6.5a}$$

$$z_k = \mathcal{C}x_k + e_k, \tag{6.5b}$$

where the actual system matrix is defined by

$$\mathcal{A}_k^a \triangleq \mathcal{A} + \mathcal{N}u_k - \mathcal{N}\tilde{u}_k. \tag{6.6}$$

As in the linear case (see (2.103a) and (2.103c)), the process noise and the output noise sequences are given by

$$v_k = \mathcal{G}w_k - \mathcal{B}\tilde{u}_k, \tag{6.7a}$$

$$e_k = \tilde{y}_k - \mathcal{D}\tilde{u}_k, \tag{6.7b}$$

respectively. The superscript a for the system matrix $\mathcal{A}_k$ is introduced here in order to distinguish between the actual matrix corresponding to the system which generates

the data and the subsequently utilised design matrix corresponding to a model which is used for Kalman filter design. Making use of different approximations of $\mathcal{A}_k^a$ leads to suboptimal Kalman filters, which are developed in Section 6.4.1 and Section 6.4.2.

6.2.2 Linear system with state dependent noise

Within this framework, one can take two different (yet mathematically equivalent) points of view. One possibility is to consider a linear time-invariant (LTI) system with state dependent process noise depending on u_{0_k}, i.e.

$$x_{k+1} = \mathcal{A}x_k + \mathcal{B}u_k + v_k^*, \tag{6.8a}$$

$$z_k = \mathcal{C}x_k + e_k, \tag{6.8b}$$

where

$$v_k^* = \mathcal{G}w_k - \mathcal{B}\tilde{u}_k + \mathcal{N}x_k u_{0_k}. \tag{6.9}$$

The noise acting on the state is now dependent on the state and the unknown input. Note that this interpretation of the bilinear EIV system requires that the input satisfies the following assumption.

AI3 The true input u_{0_k} is a stationary zero-mean ergodic process with variance σ_{u_0}.

The variance of the true input signal is either known a priori, or an estimate, denoted $\hat{\sigma}_{u_0}$, can be obtained from the data and the known variance of the input noise. A filter within this framework is derived in Section 6.4.4.

Remark 6.1. System (6.8) is a bilinear system with stochastic inputs, which is frequently referred to within the literature as a bilinear stochastic system. Such systems have many practical applications (cf. Carravetta et al. 1997). Another name for these systems is stochastic discrete-time systems with multiplicative noise, which can also be viewed as systems with stochastic parameters (depending on the point of view which is taken). Indeed, the filtering problem for such systems is addressed by making use of the interpretation of state dependent noise systems, which has been given in (6.8a)-(6.9). In the case of white stochastic system parameters, the optimal filter is discussed in (De Koning 1984) where a direct application of the standard KF is proposed, after the first- and second-order moments of the 'reformulated' non-stationary noise term v_k^* are determined. If the uncertainties are assumed to be bounded, robust filters can be developed, as outlined in (Wang & Balakrishnan 2002). Further references within these frameworks are given by (Yaz 1992, Geromel 1999).

Alternatively, by taking the second viewpoint, it is possible to consider the bilinear EIV system as a LTV system with state dependent noise based on $\tilde{u}_k$ rather than using

u_{0_k} as in (6.8a). This gives the system description

$$x_{k+1} = \mathcal{A}_k^d x_k + \mathcal{B}u_k + \bar{v}_k, \tag{6.10a}$$
$$z_k = \mathcal{C}x_k + e_k, \tag{6.10b}$$

where

$$\mathcal{A}_k^d = \mathcal{A} + \mathcal{N}u_k, \tag{6.11a}$$
$$\bar{v}_k = \mathcal{G}w_k - \mathcal{B}\tilde{u}_k - \mathcal{N}x_k\tilde{u}_k. \tag{6.11b}$$

The remainder of this chapter explores filtering techniques within both frameworks: Sections 6.4.1-6.4.2 consider a LTV system, where the system matrix is approximated in different ways, whereas Section 6.4.3 uses a LTV system with state dependent noise given by $\bar{v}_k$. Section 6.4.4 considers a LTI system with state dependent noise term given by v_k^*. A benchmark filter that is not achievable in practice is given in Section 6.3.

6.3 A benchmark filter

As for the non-EIV case, the bilinear EIV state space system can be regarded as a linear time varying system, where the system matrix depends on the unknown input u_{0_k}, i.e.

$$x_{k+1} = \mathcal{A}_k^a x_k + \mathcal{B}u_k + v_k, \tag{6.12a}$$
$$z_k = \mathcal{C}x_k + e_k, \tag{6.12b}$$

where

$$v_k = \mathcal{G}w_k - \mathcal{B}\tilde{u}_k, \tag{6.13a}$$
$$e_k = \tilde{y}_k - \mathcal{D}\tilde{u}_k, \tag{6.13b}$$
$$\mathcal{A}_k^a = \mathcal{A} + \mathcal{N}u_{0_k}. \tag{6.13c}$$

Clearly, if u_{0_k} (hence the system matrix $\mathcal{A}_k^a$) is known, one could design the EIVKF as in the linear case since the first and second order statistics of v_k and e_k are readily known and all conditions of standard Kalman filtering are fulfilled. The resulting filter is denoted BEIVKF1 and can therefore be derived by applying the EIVKF (cf. Algorithm 2.3 on page 42) using the state space model (6.12). This is achieved by setting $A_k = \mathcal{A}_k^a$, $B_k = \mathcal{B}$, $C_k = \mathcal{C}$, $D_k = \mathcal{D}$, $G_k = \mathcal{G}$, whilst the covariance matrices for

the SISO case are given by (2.105)

$$\Sigma_v^k = \Sigma_v = \mathcal{G}\sigma_w\mathcal{G}^T + \sigma_{\tilde{u}}\mathcal{B}\mathcal{B}^T, \tag{6.14a}$$

$$\sigma_e^k = \sigma_e = \sigma_{\tilde{y}} - \sigma_{\tilde{u}\tilde{y}}\mathcal{D}^T - \mathcal{D}\sigma_{\tilde{u}\tilde{y}} + \sigma_{\tilde{u}}\mathcal{D}\mathcal{D}^T, \tag{6.14b}$$

$$\Sigma_{ve}^k = \Sigma_{ve} = \mathcal{B}\sigma_{\tilde{u}}\mathcal{D}^T - \mathcal{B}\sigma_{\tilde{u}\tilde{y}}. \tag{6.14c}$$

Such a benchmark filter, which is mainly used for comparison purposes in the subsequent development, can be summarised as follows.

Algorithm 6.1 (BEIVKF1 - linear benchmark filter).

$$\hat{x}_{k+1|k} = \mathcal{A}_k^a\hat{x}_{k|k-1} + \mathcal{B}u_k + K_k\varepsilon_k \tag{6.15a}$$

$$K_k = \left[\mathcal{A}_k^a P_{k|k-1}\mathcal{C}^T + \Sigma_{ve}\right]\left[\Sigma_\varepsilon^k\right]^{-1} \tag{6.15b}$$

$$P_{k+1|k} = \mathcal{A}_k^a P_{k|k-1}\mathcal{A}_k^{a^T} + \Sigma_v - K_k\Sigma_\varepsilon^k K_k^T \tag{6.15c}$$

$$\varepsilon_k = z_k - \mathcal{C}\hat{x}_{k|k-1} \tag{6.15d}$$

$$\Sigma_\varepsilon^k = \mathcal{C}P_{k|k-1}\mathcal{C}^T + \sigma_e \tag{6.15e}$$

$$\hat{y}_{0_k} = y_k - \left[\sigma_{\tilde{y}} - \sigma_{\tilde{u}\tilde{y}}\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\varepsilon_k \tag{6.15f}$$

$$\hat{u}_{0_k} = u_k - \left[\sigma_{\tilde{u}\tilde{y}} - \sigma_{\tilde{u}}\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\varepsilon_k \tag{6.15g}$$

The expected performances for the input and output estimates of the filter are given by (2.108)

$$P_u^k = \sigma_{\tilde{u}}^k - \left[\sigma_{\tilde{u}\tilde{y}}^k - \sigma_{\tilde{u}}^k\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\left[\sigma_{\tilde{u}\tilde{y}}^k - \sigma_{\tilde{u}}^k\mathcal{D}^T\right]^T, \tag{6.16a}$$

$$P_y^k = \sigma_{\tilde{y}}^k - \left[\sigma_{\tilde{y}}^k - \sigma_{\tilde{u}\tilde{y}}^{k^T}\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\left[\sigma_{\tilde{y}}^k - \sigma_{\tilde{u}\tilde{y}}^{k\ T}\mathcal{D}^T\right]^T. \tag{6.16b}$$

Due to the very nature of the EIV problem, however, the true input u_{0_k} is not available to design the optimal EIVKF, which assumes that the true input is deterministic and known. Therefore, suboptimal alternatives are to be explored in the subsequent development. Firstly, however, to establish a benchmark a numerical example for the BEIVKF1 is presented, which shows that the filter is working well under ideal conditions.

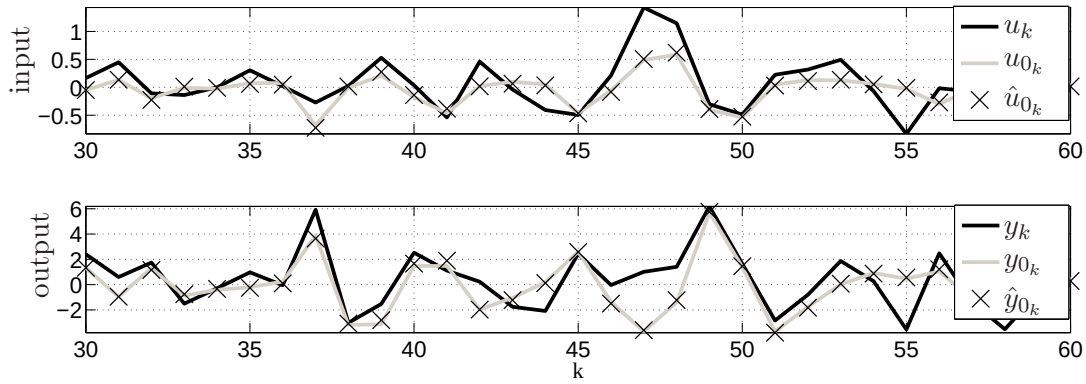


Figure 6.1: Comparison of noise-free, noisy and filtered signals using the BEIVKF1.

6.3.1 Numerical example

Consider a bilinear EIV system of the form (6.1) where the matrices are given by

$$\mathcal{A} = \begin{bmatrix} 0 & 1 \\ -0.2 & 0.3 \end{bmatrix}, \qquad \mathcal{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad \mathcal{C} = \begin{bmatrix} 0.9 & 4.05 \end{bmatrix},$$
$$\mathcal{D} = -4.5, \qquad \mathcal{N} = \begin{bmatrix} 0 & 0 \\ 0.2 & 0.5 \end{bmatrix}, \qquad \mathcal{G} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{6.17}$$

and where the noise sequences are Gaussian, zero mean and of variance and covariance

$$\sigma_w = 0.05, \qquad \sigma_{\tilde{u}} = 0.1, \qquad \sigma_{\tilde{y}} = 2.5, \qquad \sigma_{\tilde{u}\tilde{y}} = 0.5, \tag{6.18}$$

respectively. The noise-free input is chosen to be a zero mean, white Gaussian process of variance $\sigma_{u_0} = 0.1$. This corresponds to a signal-to-noise ratio on input and output of 0dB and 5.5dB, respectively. The BEIVKF1 is applied to estimate the noise-free input and output of the EIV system for $N = 100$ samples. The system is simulated with zero initial conditions whilst the initial value of the state estimation error covariance matrix is chosen to be the identity matrix.

An extract (corresponding to the time duration in samples 30-60) of the noisy, noise-free and filtered input and output signals is presented in Figure 6.1. It is observed that the noise-free input u_{0_k} as well as the noise-free output y_{0_k} can be estimated nearly perfectly. Since it is difficult to read the magnitude of the estimation errors of input and output from Figure 6.1, an extract of the estimation errors is shown in Figure 6.2. The estimation error variances of input and output, which are computed via (6.16) are also shown in Figure 6.2. It is observed that, in contrast to the LTI case, the error covariance of the input and output estimates do not tend towards a constant value. This is an expected feature since the bilinear system is considered to be a LTV system with input dependent matrix $\mathcal{A}_k^a$. Therefore, the state estimation error covariance matrix $P_{k|k-1}$ is a time varying quantity. This property is propagated to the innovations covariance

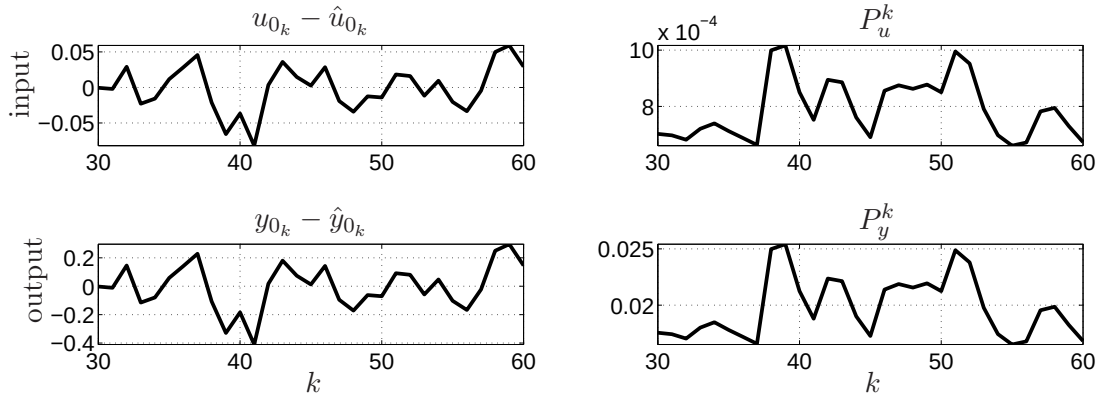


Figure 6.2: An extract of the estimation error of input and output estimates and their corresponding variances computed via (6.16).

Σ_ε^k, which finally results into time varying variances of input and output estimation errors (cf. (6.16)).

In addition to the error variances, another convenient measure for the performance of the EIV filter may be given by

$$M_u = 100\frac{||u_0 - u||_2 - ||u_0 - \hat{u}_0||_2}{||u_0 - u||_2}, \tag{6.19a}$$

$$M_y = 100\frac{||y_0 - y||_2 - ||y_0 - \hat{y}_0||_2}{||y_0 - y||_2}, \tag{6.19b}$$

where the variables without time index (u, y, etc.) denote the corresponding sequence from 1 to N, with N being the number of samples. These performance measures can be considered as the 'amount' of noise in percentage, which is removed by the filter from the noisy input and output signals, respectively. A negative value would indicate that the filter is not working satisfactorily, i.e. that, on average, the noisy signal is 'closer' to the 'true' signal than its estimation. Note that the performance measures (6.19) are purely of an academic nature, since u_0 and y_0 would not be known in practice. For the given example the values of M_u and M_y are virtually identical and given by 90.7. This means that about 90% of the noise can be removed from both the output and input signals.

Remark 6.2. The results of the previous example might appear surprisingly good to the reader, however, the conditions within the example are, for illustrative purposes, carefully chosen: Firstly, the linear part of system (6.17) has a unity steady state gain, which allows an approximately equal performance for input and output estimates to be obtained. Furthermore, the variances of the noise sequences have quite a significant impact on the filter performance. For example, when $\sigma_{\tilde{u}\tilde{y}}$ in (6.18) is altered from 0.5 to 0.2 and the whole experiment is repeated, the filter performance, specified by (6.19), degrades from approximately 90% to 35%. These aspects should be taken into consideration, if the EIV filter is aimed to be applied to a practical system. Whilst the

system within the current example is of a bilinear structure, similar experiences have been made with the linear EIVKF, which has been reviewed in Section 2.5.2.

6.4 Development of suboptimal filters

6.4.1 Linear Kalman filter design using u_k

Since u_{0_k} is unknown, a pragmatic approach for the design of a linear Kalman filter can then be taken by ignoring the uncertainty $\mathcal{N}\tilde{u}_k$, which is acting on the actual system matrix $\mathcal{A}_k^a = \mathcal{A} + \mathcal{N}u_k - \mathcal{N}\tilde{u}_k$. In the case of high signal to noise ratio on the input, this might appear to be a reasonable choice. The corresponding system matrix for the Kalman filter design is given by

$$\mathcal{A}_k^d = \mathcal{A} + \mathcal{N}u_k, \tag{6.20}$$

whilst the covariance matrices of the noise terms remains as presented in (6.14). Since the signal model differs from the actual system, the corresponding KF, which is denoted BEIVKF2, is inevitably suboptimal and can be summarised as follows.

Algorithm 6.2 (BEIVKF2 - linear suboptimal filter).

$$\hat{x}_{k+1|k} = \mathcal{A}_k^d \hat{x}_{k|k-1} + \mathcal{B}u_k + \bar{K}_k \bar{\varepsilon}_k \tag{6.21a}$$

$$\bar{K}_k = \left[\mathcal{A}_k^d \bar{P}_{k|k-1}\mathcal{C}^T + \Sigma_{ve}\right]\left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1} \tag{6.21b}$$

$$\bar{P}_{k+1|k} = \mathcal{A}_k^d \bar{P}_{k|k-1}\mathcal{A}_k^{d^T} + \Sigma_v - \bar{K}_k \Sigma_{\bar{\varepsilon}}^k \bar{K}_k^T \tag{6.21c}$$

$$\bar{\varepsilon}_k = z_k - \mathcal{C}\hat{x}_{k|k-1} \tag{6.21d}$$

$$\Sigma_{\bar{\varepsilon}}^k = \mathcal{C}\bar{P}_{k|k-1}\mathcal{C}^T + \sigma_e \tag{6.21e}$$

$$\hat{y}_{0_k} = y_k - \left[\sigma_{\tilde{y}} - \sigma_{\tilde{u}\tilde{y}}\mathcal{D}^T\right]\left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1}\bar{\varepsilon}_k \tag{6.21f}$$

$$\hat{u}_{0_k} = u_k - \left[\sigma_{\tilde{u}\tilde{y}} - \sigma_{\tilde{u}}\mathcal{D}^T\right]\left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1}\bar{\varepsilon}_k \tag{6.21g}$$

Corresponding to (6.16), one obtains for the suboptimal filter

$$\bar{P}_u^k = \sigma_{\tilde{u}}^k - \left[\sigma_{\tilde{u}\tilde{y}}^k - \sigma_{\tilde{u}}^k\mathcal{D}^T\right]\left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1}\left[\sigma_{\tilde{u}\tilde{y}}^k - \sigma_{\tilde{u}}^k\mathcal{D}^T\right]^T, \tag{6.22a}$$

$$\bar{P}_y^k = \sigma_{\tilde{y}}^k - \left[\sigma_{\tilde{y}}^k - \sigma_{\tilde{u}\tilde{y}}^{kT}\mathcal{D}^T\right]\left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1}\left[\sigma_{\tilde{y}}^k - \sigma_{\tilde{u}\tilde{y}}^{k\ T}\mathcal{D}^T\right]^T. \tag{6.22b}$$

The uncertainty in the design matrix $\mathcal{A}_k^d$, given by $\mathcal{N}\tilde{u}_k$, can be interpreted as a signal model error. The fact that the signal model used for filtering is imperfect is quite common for practical filter design (Anderson & Moore 1979, Jazwinski 1970), since

almost always the model is only an approximation of an unknown real-world system. In general, these model uncertainties can be due to identification errors, linearisation and/or other approximations, whereas it arises here explicitly due to the input measurement noise $\tilde{u}_k$. Since the model used for filter design differs from the model description (6.1), the corresponding KF is no longer the minimum variance estimator, nor is $\bar{P}_{k+1|k}$ the estimation error covariance matrix (Jazwinski 1970). This means that $\bar{P}_{k+1|k}$ provides 'false' information about the actual quality of the state estimates, hence in the case of the EIV configuration, about the quality of the estimates of the noise-free inputs and outputs (which are computed based on $\bar{P}_{k+1|k}$). In the worst case, this may even lead to divergence of the filter. However, it is possible to analyse the qualitative effects of such model errors as outlined in (Jazwinski 1970). For the BEIVKF2, such an analysis is carried out in the following subsection.

Analysis of filter performance

The degradation of the filter performance is now analysed following the approach given in (Jazwinski 1970, p. 244). In order to proceed, an additional assumption is required.

AS4 The bilinearity $\mathcal{N}$ is chosen such that y_{0_k} is zero mean.

Note that in the case of bilinear systems the zero mean property of the output, which is stated by **AS4**, is not guaranteed by choosing a zero mean input, as in the linear case (Pearson 1999). Assumption **AS4** ensures that the (unconditional) mean of the state is zero, a property which will be exploited in the subsequent development.

Recall that the actual system is defined by

$$x_{k+1} = \mathcal{A}_k^a x_k + \mathcal{B}u_k + v_k, \tag{6.23a}$$

$$z_k = \mathcal{C}x_k + e_k, \tag{6.23b}$$

whereas the model for filter design is given by

$$\bar{x}_{k+1} = \mathcal{A}_k^d \bar{x}_k + \mathcal{B}u_k + v_k, \tag{6.24a}$$

$$\bar{z}_k = \mathcal{C}\bar{x}_k + e_k, \tag{6.24b}$$

where the bar-notation is used to distinguish between the state and output of the system and the design model. The corresponding KF is given by (6.15a)-(6.15e). Note that $\bar{K}_k$ is not the optimal Kalman gain, nor is $\bar{P}_{k+1|k}$ the estimation error covariance matrix (as outlined above), since the model differs from the actual system. Consequently, (6.15a)-(6.15e) is not the minimum variance filter for (6.23). Using $\mathcal{A}_k^a = \mathcal{A} + \mathcal{N}u_{0_k}$,

the actual estimation error is given by

$$\begin{aligned}\tilde{x}_{k+1} &\triangleq x_{k+1} - \hat{x}_{k+1|k} \\ &= \mathcal{A}_k^a x_k + \mathcal{B}u_k + v_k - \left[\mathcal{A}_k^d \hat{x}_{k|k-1} + \mathcal{B}u_k + \bar{K}_k \left[z_k - \mathcal{C}\hat{x}_{k|k-1}\right]\right] \\ &= \mathcal{A}_k^a x_k + \mathcal{G}w_k - \mathcal{B}\tilde{u}_k - \mathcal{A}_k^a \hat{x}_{k|k-1} - \mathcal{N}\tilde{u}_k \hat{x}_{k|k-1} - \bar{K}_k \left[\mathcal{C}\tilde{x}_k + e_k\right] \\ &= \left[\mathcal{A} - \bar{K}_k\mathcal{C}\right]\tilde{x}_k + \mathcal{N}u_{0_k}\tilde{x}_k + \mathcal{G}w_k - \mathcal{N}\tilde{u}_k x_k + \mathcal{N}\tilde{u}_k\tilde{x}_k - \bar{K}_k\tilde{y}_k + \left[\bar{K}_k\mathcal{D} - \mathcal{B}\right]\tilde{u}_k.\end{aligned} \tag{6.25}$$

The third equality makes use of (6.23b), whilst the fourth equality utilises (6.13a) and (6.4b). A measure for the filter performance is then given by the actual estimation error covariance matrix, which is given by

$$P_{k+1|k} \triangleq E\left[\tilde{x}_{k+1}\tilde{x}_{k+1}^T\right]. \tag{6.26}$$

Due to the mutual independence of $\tilde{x}_k$, w_k, $\tilde{u}_k$ and $\tilde{y}_k$, and since $\tilde{u}_k$ is zero mean as well as independent of x_k, the estimation error covariance matrix is given by

$$\begin{aligned}P_{k+1|k} = {} & \left[\mathcal{A} - \bar{K}_k\mathcal{C}\right] P_{k|k-1} \left[\mathcal{A} - \bar{K}_k\mathcal{C}\right]^T + \mathcal{N}\sigma_{u_0} P_{k|k-1}\mathcal{N}^T + \mathcal{G}\sigma_w\mathcal{G}^T \\ & + \mathcal{N}\sigma_{\tilde{u}}\Sigma_x^k\mathcal{N}^T - \mathcal{N}\sigma_{\tilde{u}}\Sigma_{x\tilde{x}}^k\mathcal{N}^T \\ & - \mathcal{N}\sigma_{\tilde{u}}\Sigma_{x\tilde{x}}^k\mathcal{N}^T + \mathcal{N}\sigma_{\tilde{u}}P_{k|k-1}\mathcal{N}^T \\ & + \bar{K}_k\sigma_{\tilde{y}}\bar{K}_k^T - \bar{K}_k\sigma_{\tilde{u}\tilde{y}}\left[\bar{K}_k\mathcal{D} - \mathcal{B}\right]^T \\ & - \left[\bar{K}_k\mathcal{D} - \mathcal{B}\right]\sigma_{\tilde{u}\tilde{y}}\bar{K}_k^T + \left[\bar{K}_k\mathcal{D} - \mathcal{B}\right]\sigma_{\tilde{u}}\left[\bar{K}_k\mathcal{D} - \mathcal{B}\right]^T,\end{aligned} \tag{6.27}$$

where

$$\Sigma_x^k \triangleq E\left[x_k x_k^T\right], \tag{6.28a}$$

$$\Sigma_{x\tilde{x}}^k \triangleq E\left[x_k\tilde{x}_k^T\right]. \tag{6.28b}$$

Note that for the computation of (6.27), the property of zero mean input and output as well as zero mean state (unconditional) and state estimation error has been utilised (see **AS4**). From (6.1a) and (6.25), recursive expressions of Σ_x^k and $\Sigma_{x\tilde{x}}^k$ can be obtained as

$$\Sigma_x^{k+1} = \mathcal{A}\Sigma_x^k\mathcal{A}^T + \mathcal{B}\sigma_{u_0}\mathcal{B}^T + \mathcal{N}\sigma_{u_0}\Sigma_x^k\mathcal{N}^T + \mathcal{G}\sigma_w\mathcal{G}^T, \tag{6.29a}$$

$$\Sigma_x^0 = P_0 + \bar{x}_0\bar{x}_0^T \tag{6.29b}$$

and

$$\Sigma_{x\tilde{x}}^{k+1} = \mathcal{A}\Sigma_{x\tilde{x}}^k\left[\mathcal{A} - \bar{K}_k\mathcal{C}\right]^T + \mathcal{N}\sigma_{u_0}\Sigma_{x\tilde{x}}^k\mathcal{N}^T + \mathcal{G}\sigma_w\mathcal{G}^T, \tag{6.30a}$$

$$\Sigma_{x\tilde{x}}^0 = P_0, \tag{6.30b}$$

respectively. Finally, if not known a priori, the second order moments of the noise-free input and noise-free output can be obtained via

$$\sigma_{u0} \triangleq E\left[u_{0_k}^2\right] = E\left[u_k^2 - 2u_k\tilde{u}_k + \tilde{u}_k^2\right] = \sigma_u - \sigma_{\tilde{u}}, \tag{6.31a}$$

$$\sigma_{y0} \triangleq E\left[y_{0_k}^2\right] = E\left[y_k^2 - 2y_k\tilde{y}_k + \tilde{y}_k^2\right] = \sigma_y - \sigma_{\tilde{y}}, \tag{6.31b}$$

with $E[u_k^2] = \sigma_u$ and $E[y_k^2] = \sigma_y$. The latter quantities can be estimated recursively from the available measurements as

$$\hat{\sigma}_u^k = \hat{\sigma}_u^{k-1} + \frac{1}{k}\left[u_k^2 - \hat{\sigma}_u^{k-1}\right], \tag{6.32}$$

$$\hat{\sigma}_y^k = \hat{\sigma}_y^{k-1} + \frac{1}{k}\left[y_k^2 - \hat{\sigma}_y^{k-1}\right]. \tag{6.33}$$

By making use of (6.14) and (6.31), Equation (6.27) can be further simplified which finally gives

$$\begin{aligned} P_{k+1|k} = {} & \left[\mathcal{A} - \bar{K}_k\mathcal{C}\right] P_{k|k-1} \left[\mathcal{A} - \bar{K}_k\mathcal{C}\right]^T + \Sigma_v + \bar{K}_k\sigma_e\bar{K}_k^T - \Sigma_{ve}\bar{K}_k^T - \bar{K}_k\Sigma_{ve}^T \\ & + \mathcal{N}\sigma_u P_{k|k-1}\mathcal{N}^T + \mathcal{N}\sigma_{\tilde{u}}\Sigma_x^k\mathcal{N}^T - 2\mathcal{N}\sigma_{\tilde{u}}\Sigma_{x\tilde{x}}^k\mathcal{N}^T. \end{aligned} \tag{6.34}$$

In addition, by introducing the actual variance of the innovations as

$$\Sigma_\varepsilon^k = \mathcal{C}P_{k|k-1}\mathcal{C}^T + \sigma_e, \tag{6.35}$$

the actual variances of the input and the output estimation error (in contrast to the 'false' quantities (6.22)) are given by

$$P_u^k = \sigma_{\tilde{u}}^k - \left[\sigma_{\tilde{u}\tilde{y}}^k - \sigma_{\tilde{u}}^k\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\left[\sigma_{\tilde{u}\tilde{y}}^k - \sigma_{\tilde{u}}^k\mathcal{D}^T\right]^T, \tag{6.36a}$$

$$P_y^k = \sigma_{\tilde{y}}^k - \left[\sigma_{\tilde{y}}^k - \sigma_{\tilde{u}\tilde{y}}^{kT}\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\left[\sigma_{\tilde{y}}^k - \sigma_{\tilde{u}\tilde{y}}^{k\ T}\mathcal{D}^T\right]^T. \tag{6.36b}$$

The values of $P_{k+1|k}$, P_u^k and P_y^k indicate how satisfactorily the filter is performing, which can be utilised in practice to make a decision on whether or not to apply the filter.

There exist, however, alternative approaches to resolve the bilinear EIV filtering problem, some of which are addressed in the following subsections. Before proceeding, a numerical simulation is provided, which aims to validate the preceding error analysis. A comparison of the filter performance is postponed until Section 6.5, where not only the BEIVKF2, but also the other filters, which are developed within Sections 6.4.2, 6.4.3 and 6.4.4 are compared with the BEIVKF1.

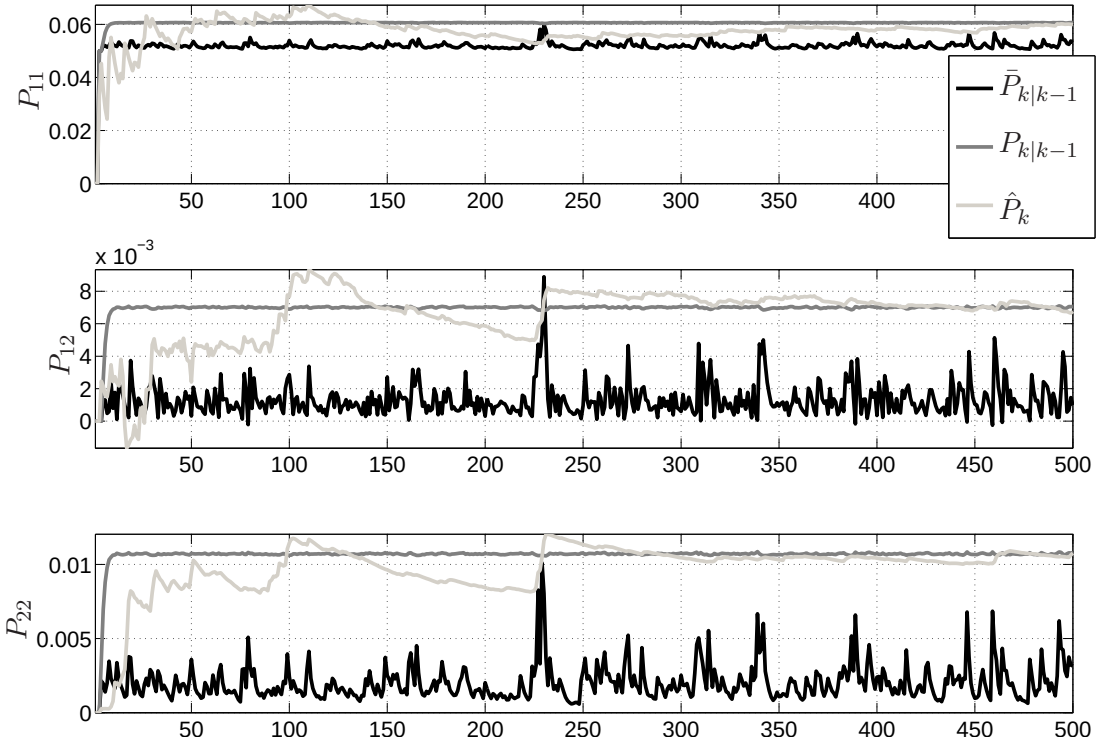


Figure 6.3: Comparison of ('false') error covariance matrix $\bar{P}_{k|k-1}$ computed by the BEIVKF2 with the actual value of $P_{k|k-1}$ and a sample estimate $\hat{P}_k$.

Numerical example

Consider the setup given in Section 6.3.1. The suboptimal BEIVKF2 is applied to estimate the system state vector and the noise-free input and noise-free output. The ('false') error covariance matrix $\bar{P}_{k|k-1}$ (6.21c) is compared with the actual error covariance matrix $P_{k|k-1}$, which can be computed by (6.34). In order to validate the results, these quantities could be compared with a sample estimate of the error covariance matrix, which is obtained via

$$\hat{P}_k \triangleq \frac{1}{k}\sum_{i=1}^{k} \tilde{x}_i \tilde{x}_i^T. \tag{6.37}$$

It is noted, however, that the computation of $\hat{P}_k$ assumes ergodicity and stationarity of $\tilde{x}_k$, which cannot be guaranteed due to the time varying nature of the bilinear state space system. The values of $\hat{P}_k$ are therefore only of an indicative character and can be regarded as a rough approximation only. The results for a particular realisation using $N = 500$ samples are displayed in Figure 6.3. It is observed that the estimation error covariance computed by the BEIVKF2 appears to be over optimistic: Considering the diagonal elements P_{11} and P_{22}, the actual values corresponding to $P_{k|k-1}$ are larger than those of $\bar{P}_{k|k-1}$, which is computed by the BEIVKF2. The values of $P_{k|k-1}$ seem

to be in broad agreement with those of the sample covariance $\hat{P}_k$, which appears to underpin the theoretical results obtained in this section. The peaks which are observed around a value of $k = 230$, might be due to the fact that the stability assumption of the bilinear system (**AI4**) is violated, which means that the BEIVKF2 becomes temporarily unstable (recall that the input is drawn from a zero mean Gaussian distribution).

It can be concluded that the BEIVKF2 algorithm provides false information about the state estimation accuracy due to the uncertainty in the model, which is used for the Kalman filter design (and which is basically due to the noise on the input). Note that this also applies to the error covariance matrices of the noise-free input and output estimates $\bar{P}_u^k$ and $\bar{P}_y^k$, given in (6.22), since these quantities are dependent on $\bar{P}_{k|k-1}$.

6.4.2 Nonlinear Kalman filter design using $\hat{u}_{0_k}$

Since the filtered input $\hat{u}_{0_k}$ at time k can be evaluated before the Kalman recursions (i.e. the computation of $\hat{x}_{k|k-1}$, $P_{k|k-1}$ and K_k) take place, it appears to be most natural to utilise $\hat{u}_{0_k}$ in order to (hopefully) better approximate $\mathcal{A}_k^a$ in (6.15a)-(6.15c). Such a cross-coupling of the estimate and filter design is commonly used within the extended Kalman filter (EKF). This leads to the signal model

$$\bar{x}_{k+1} = \hat{\mathcal{A}}_k \bar{x}_k + \mathcal{B} u_k + v_k, \tag{6.38a}$$

$$\bar{z}_k = \mathcal{C} \bar{x}_k + e_k, \tag{6.38b}$$

where

$$v_k = \mathcal{G} w_k - \mathcal{B} \tilde{u}_k, \tag{6.39a}$$

$$e_k = \tilde{y}_k - \mathcal{D} \tilde{u}_k, \tag{6.39b}$$

$$\hat{\mathcal{A}}_k = \mathcal{A} + \mathcal{N} \hat{u}_{0_k}, \tag{6.39c}$$

and where the corresponding covariance matrices are given by (6.14). The corresponding EIVKF, which is denoted BEIVKF3, is obtained by substituting $\mathcal{A}_k^a \approx \hat{\mathcal{A}}_k$ in the BEIVKF1 algorithm, and can be summarised as follows.

Algorithm 6.3 (BEIVKF3).

$$\hat{x}_{k+1|k} = \hat{\mathcal{A}}_k \hat{x}_{k|k-1} + \mathcal{B} u_k + \bar{K}_k \bar{\varepsilon}_k \tag{6.40a}$$

$$\bar{K}_k = \left[\hat{\mathcal{A}}_k \bar{P}_{k|k-1} \mathcal{C}^T + \Sigma_{ve} \right] \left[\Sigma_{\bar{\varepsilon}}^k \right]^{-1} \tag{6.40b}$$

$$\bar{P}_{k+1|k} = \hat{\mathcal{A}}_k \bar{P}_{k|k-1} \hat{\mathcal{A}}_k^T + \Sigma_v - \bar{K}_k \Sigma_{\bar{\varepsilon}}^k \bar{K}_k^T \tag{6.40c}$$

$$\bar{\varepsilon}_k = z_k - \mathcal{C} \hat{x}_{k|k-1} \tag{6.40d}$$

$$\Sigma_{\bar{\varepsilon}}^k = \mathcal{C} \bar{P}_{k|k-1} \mathcal{C}^T + \sigma_e \tag{6.40e}$$

$$\hat{y}_{0_k} = y_k - \left[\sigma_{\tilde{y}} - \sigma_{\tilde{u}\tilde{y}}\mathcal{D}^T\right]\left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1}\bar{\varepsilon}_k \tag{6.40f}$$

$$\hat{u}_{0_k} = u_k - \left[\sigma_{\tilde{u}\tilde{y}} - \sigma_{\tilde{u}}\mathcal{D}^T\right]\left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1}\bar{\varepsilon}_k \tag{6.40g}$$

$$\hat{\mathcal{A}}_k = \mathcal{A} + \mathcal{N}\hat{u}_{0_k} \tag{6.40h}$$

Remark 6.3 (Error analysis). Whilst for the BEIVKF2, the uncertainty in the system matrix (6.20) is characterised by the input measurement noise $\tilde{u}_k$ with a priori known covariance $\sigma_{\tilde{u}}$, the uncertainty in $\hat{\mathcal{A}}_k$ can be quantified by the estimation error $u_{0_k} - \hat{u}_{0_k}$, hence ultimately by the state estimation error $\tilde{x}_k$. As in the case of the BEIVKF2 algorithm, the error covariance matrix produced by the BEIVKF3 provides incorrect information about the actual filter performance. Hence, one could perform an error analysis as for the BEIVKF2. This leads, however, to a rather complex expression for the actual estimation error covariance matrix which also involves higher order moments of the state estimation error. An expression for the state estimation error is given in Appendix I. Whilst a thorough analysis is identified as potential further work, it is not considered within this thesis.

6.4.3 Nonlinear Kalman filter design using $\hat{x}_{k|k-1}$

Whilst in Section 6.4.2 the estimate of the noise-free input has been cross-coupled with the design of the BEIVKF, a cross-coupling of the current state estimate and BEIVKF design seems also possible. Therefore, the bilinear EIV system (6.3) is interpreted as a linear time varying system with known system matrix and state dependent noise, given by

$$x_{k+1} = \mathcal{A}_k^d x_k + \mathcal{B}u_k + \bar{v}_k, \tag{6.41a}$$

$$z_k = \mathcal{C}x_k + e_k, \tag{6.41b}$$

where

$$\mathcal{A}_k^d = \mathcal{A} + \mathcal{N}u_k, \tag{6.42a}$$

$$e_k = \tilde{y}_k - \mathcal{D}\tilde{u}_k, \tag{6.42b}$$

$$\bar{v}_k = \mathcal{G}w_k - \mathcal{B}\tilde{u}_k - \mathcal{N}x_k\tilde{u}_k. \tag{6.42c}$$

The state-dependent noise term $\bar{v}_k$ can be re-expressed as

$$\bar{v}_k = \mathcal{G}_k^a v_k', \tag{6.43}$$

with

$$\mathcal{G}_k^a \triangleq \begin{bmatrix} \mathcal{G} & \mathcal{B} + \mathcal{N} x_k \end{bmatrix}, \tag{6.44a}$$

$$v_k' \triangleq \begin{bmatrix} w_k \\ -\tilde{u}_k \end{bmatrix}, \tag{6.44b}$$

where $\mathcal{G}_k^a$ is now viewed as a state-dependent, hence uncertain, matrix. It seems natural to utilise the approximation

$$\mathcal{G}_k^d = \begin{bmatrix} \mathcal{G} & \mathcal{B} + \mathcal{N} \hat{x}_{k|k-1} \end{bmatrix} \tag{6.45}$$

as the design matrix for the BEIVKF, which is assumed to be known at time k. The covariances of the approximated state noise are readily given by

$$\Sigma_{\bar{v}}^k = \mathcal{G}_k^d E\left[v_k' {v_k'}^T\right] \mathcal{G}_k^{dT} = \mathcal{G}_k^d \begin{bmatrix} \sigma_w & 0 \\ 0 & \sigma_{\tilde{u}} \end{bmatrix} \mathcal{G}_k^{dT}, \tag{6.46a}$$

$$\Sigma_{\bar{v}e}^k = \mathcal{G}_k^d E\left[v_k' e_k^T\right] = \mathcal{G}_k^d \begin{bmatrix} 0 \\ \sigma_{\tilde{u}} \mathcal{D}^T - \sigma_{\tilde{u}\tilde{y}} \end{bmatrix}, \tag{6.46b}$$

and a KF can be applied. Note that such an approach could be interpreted as an EKF (Anderson & Moore 1979, cf. p. 194), where $\mathcal{G}_k^a$ is approximated by zero order Taylor approximation around the latest state estimate $\hat{x}_{k|k-1}$ (i.e. x_k is simply replaced by $\hat{x}_{k|k-1}$). The resulting filter, which is denoted BEIVKF4, is suboptimal and the filter equations are nonlinear in $\hat{x}_{k|k-1}$. The BEIVKF4 algorithm can be summarised as follows.

Algorithm 6.4 (BEIVKF4).

$$\hat{x}_{k+1|k} = \mathcal{A}_k^d \hat{x}_{k|k-1} + \mathcal{B} u_k + \bar{K}_k \bar{\varepsilon}_k \tag{6.47a}$$

$$\bar{K}_k = \left[\mathcal{A}_k^d \bar{P}_{k|k-1} \mathcal{C}^T + \Sigma_{\bar{v}e}^k\right] \left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1} \tag{6.47b}$$

$$\bar{P}_{k+1|k} = \mathcal{A}_k^d \bar{P}_{k|k-1} \mathcal{A}_k^{dT} + \Sigma_{\bar{v}}^k - \bar{K}_k \Sigma_{\bar{\varepsilon}}^k \bar{K}_k^T \tag{6.47c}$$

$$\bar{\varepsilon}_k = z_k - \mathcal{C} \hat{x}_{k|k-1} \tag{6.47d}$$

$$\Sigma_{\bar{\varepsilon}}^k = \mathcal{C} \bar{P}_{k|k-1} \mathcal{C}^T + \sigma_e \tag{6.47e}$$

$$\hat{y}_{0_k} = y_k - \left[\sigma_{\tilde{y}} - \sigma_{\tilde{u}\tilde{y}} \mathcal{D}^T\right] \left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1} \bar{\varepsilon}_k \tag{6.47f}$$

$$\hat{u}_{0_k} = u_k - \left[\sigma_{\tilde{u}\tilde{y}} - \sigma_{\tilde{u}} \mathcal{D}^T\right] \left[\Sigma_{\bar{\varepsilon}}^k\right]^{-1} \bar{\varepsilon}_k \tag{6.47g}$$

$$\Sigma_{\bar{v}}^k = \begin{bmatrix} \mathcal{G} & \mathcal{B} + \mathcal{N} \hat{x}_{k|k-1} \end{bmatrix} \begin{bmatrix} \sigma_w & 0 \\ 0 & \sigma_{\tilde{u}} \end{bmatrix} \begin{bmatrix} \mathcal{G} & \mathcal{B} + \mathcal{N} \hat{x}_{k|k-1} \end{bmatrix}^T \tag{6.47h}$$

$$\Sigma_{\bar{v}e}^k = \begin{bmatrix} \mathcal{G} & \mathcal{B} + \mathcal{N} \hat{x}_{k|k-1} \end{bmatrix} \begin{bmatrix} 0 & \sigma_{\tilde{u}} \mathcal{D} - \sigma_{\tilde{u}\tilde{y}} \end{bmatrix}^T \tag{6.47i}$$

6.4.4 Kalman filter for state dependent noise system

In (De Koning 1984), the optimal state estimator in the minimum variance sense is derived for linear discrete-time systems where the system matrices contain stochastic parameters which are statistically independent with respect to time. Basically, the optimal filter is obtained by considering the system as a linear system with deterministic matrices and state-dependent noise terms, which can be replaced by a process having the same first- and second order properties. This means that if the covariance matrix of the state dependent noise term can be computed (and if it satisfies the zero mean condition), the optimal filter can be derived by applying standard linear estimation theory. Although the setup considered in (De Koning 1984) is slightly different to the bilinear EIV setup (since it assumes that the disturbances in the system matrices are uncorrelated with the noise acting on the states), the approach can be used to address the bilinear EIV filtering problem.

The bilinear EIV system is expressed as (6.8a)

$$x_{k+1} = \mathcal{A}x_k + \mathcal{B}u_k + v_k^*, \tag{6.48a}$$

$$z_k = \mathcal{C}x_k + e_k, \tag{6.48b}$$

with

$$e_k = \tilde{y}_k - \mathcal{D}\tilde{u}_k, \tag{6.49a}$$

$$v_k^* = \mathcal{G}w_k - \mathcal{B}\tilde{u}_k + \mathcal{N}x_k u_{0_k}. \tag{6.49b}$$

Whilst σ_e is given in (6.14b), the covariance matrix of the state dependent noise term can be computed as

$$\begin{aligned}\Sigma_{v^*}^k &\triangleq E\left[v_k^* v_k^{*T}\right] \\ &= E\left[\left[\mathcal{G}w_k - \mathcal{B}\tilde{u}_k + \mathcal{N}x_k u_{0_k}\right]\left[\mathcal{G}w_k - \mathcal{B}\tilde{u}_k + \mathcal{N}x_k u_{0_k}\right]^T\right] \\ &= \mathcal{G}\sigma_w\mathcal{G}^T + \mathcal{B}\sigma_{\tilde{u}}\mathcal{B}^T + \mathcal{N}\sigma_{u_0}\Sigma_x^k\mathcal{N}^T,\end{aligned} \tag{6.50}$$

where the property of zero (unconditional) mean of the state has been exploited. Note that the term Σ_x^k has already been computed for the error analysis of the BEIVKF2 algorithm given in (6.29) and that σ_{u_0} can be computed using (6.31), if it is not known a priori. It remains to compute the quantity $\Sigma_{v^*e}^k$ which is given by

$$\begin{aligned}\Sigma_{v^*e}^k &\triangleq E\left[v_k^* e_k\right] \\ &= E\left[\left[\mathcal{G}w_k - \mathcal{B}\tilde{u}_k + \mathcal{N}x_k u_{0_k}\right]\left[\tilde{y}_k - \mathcal{D}\tilde{u}_k\right]^T\right] \\ &= \mathcal{B}\sigma_{\tilde{u}}\mathcal{D}^T - \mathcal{B}\sigma_{\tilde{u}\tilde{y}},\end{aligned} \tag{6.51}$$

where the zero mean property of the state has been exploited. Note that (6.51) is

identical to Σ_{ve} given in (6.14c). Since all conditions for the application of the KF are satisfied, the optimal linear filter for state dependent noise can be given as follows.

Algorithm 6.5 (BEIVKF5).

$$\hat{x}_{k+1|k} = \mathcal{A}\hat{x}_{k|k-1} + \mathcal{B}u_k + K_k\varepsilon_k \tag{6.52a}$$

$$K_k = \left[\mathcal{A}P_{k|k-1}\mathcal{C}^T + \Sigma_{ve}\right]\left[\Sigma_\varepsilon^k\right]^{-1} \tag{6.52b}$$

$$P_{k+1|k} = \mathcal{A}P_{k|k-1}\mathcal{A}^T + \Sigma_{v^*}^k - K_k\Sigma_\varepsilon^k K_k^T \tag{6.52c}$$

$$\varepsilon_k = z_k - \mathcal{C}\hat{x}_{k|k-1} \tag{6.52d}$$

$$\Sigma_\varepsilon^k = \mathcal{C}P_{k|k-1}\mathcal{C}^T + \sigma_e \tag{6.52e}$$

$$\hat{y}_{0_k} = y_k - \left[\sigma_{\tilde{y}} - \sigma_{\tilde{u}\tilde{y}}\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\varepsilon_k \tag{6.52f}$$

$$\hat{u}_{0_k} = u_k - \left[\sigma_{\tilde{u}\tilde{y}} - \sigma_{\tilde{u}}\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\varepsilon_k \tag{6.52g}$$

$$\Sigma_{v^*}^k = \mathcal{G}\sigma_w\mathcal{G}^T + \mathcal{B}\sigma_{\tilde{u}}\mathcal{B}^T + \mathcal{N}\sigma_{u_0}\Sigma_x^k\mathcal{N}^T \tag{6.52h}$$

$$\Sigma_x^{k+1} = \mathcal{A}\Sigma_x^k\mathcal{A}^T + \mathcal{B}\sigma_{u_0}\mathcal{B}^T + \mathcal{N}\sigma_{u_0}\Sigma_x^k\mathcal{N}^T + \mathcal{G}\sigma_w\mathcal{G}^T \tag{6.52i}$$

Remark 6.4 (Optimality of the BEIVKF5). The BEIVKF5 is the optimal filter if the true input u_{0_k} is considered to be process noise with variance σ_{u_0}. However, since the input is measured, there is more available information about u_{0_k} than its first and second order moments only. This means whilst being optimal if only the first and second order moments of u_{0_k} are utilised, the BEIVKF5 does not fully exploit the knowledge of the measurements u_k, in order to estimate the state of the bilinear EIV system. Therefore, the BEIVKF5 is assumed to perform suboptimally when applied to the bilinear EIV filtering problem.

6.5 Numerical example

Consider the setup given in Section 6.3.1. The system is simulated in 100 Monte-Carlo simulations, each comprising $N = 1000$ data samples. All five filters are applied to estimate the noise-free input and output signals. For each Monte-Carlo run, the mean values of P_u^k, P_y^k and/or $\bar{P}_u^k$, $\bar{P}_y^k$ are stored. In addition, measures for the variances of the estimation errors are estimated from the data as

$$\hat{P}_u = \frac{1}{800}\sum_{i=201}^{1000}\left(u_{0_k} - \hat{u}_{0_k}\right)^2, \tag{6.53a}$$

$$\hat{P}_y = \frac{1}{800}\sum_{i=201}^{1000}\left(y_{0_k} - \hat{y}_{0_k}\right)^2, \tag{6.53b}$$

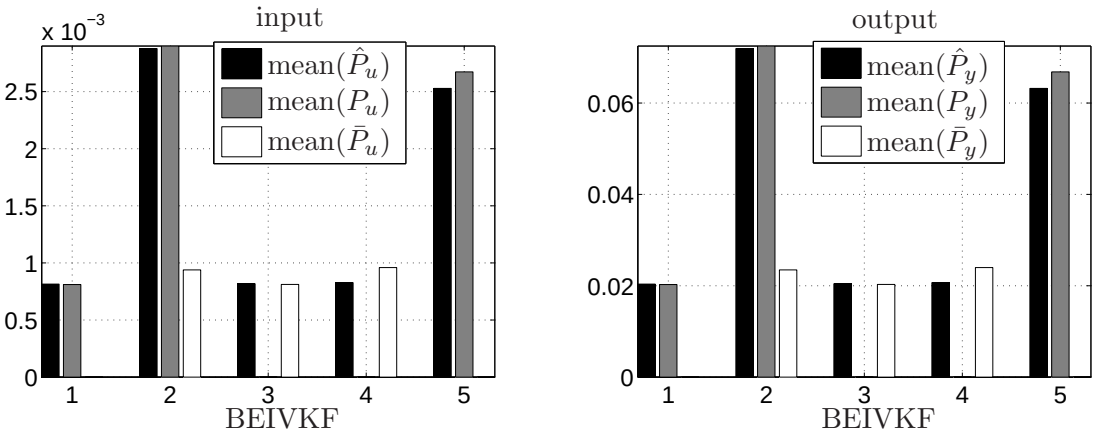


Figure 6.4: Monte-Carlo mean values of the estimation error variances for all five filters.

and stored for each Monte-Carlo run. The corresponding mean values and standard deviations of P_u^k, P_y^k, $\bar{P}_u^k$ $\bar{P}_y^k$, $\hat{P}_u$ and $\hat{P}_y$ for the five filters are given in Table 6.1. The mean values are also illustrated in Figure 6.4. It is observed that the BEIVKF1

Filter	mean($\hat{P}_u$)		mean(P_u)		mean($\bar{P}_u$)	
BEIVKF1	$8.14{\cdot}10^{-4}$	$\pm\ 4.63{\cdot}10^{-5}$	$8.11{\cdot}10^{-4}$	$\pm\ 5.44{\cdot}10^{-19}$	-	
BEIVKF2	$2.88{\cdot}10^{-3}$	$\pm\ 1.59{\cdot}10^{-4}$	$2.90{\cdot}10^{-3}$	$\pm\ 1.58{\cdot}10^{-4}$	$9.39{\cdot}10^{-4}$	$\pm\ 1.24{\cdot}10^{-5}$
BEIVKF3	$8.17{\cdot}10^{-4}$	$\pm\ 4.91{\cdot}10^{-5}$	-		$8.11{\cdot}10^{-4}$	$\pm\ 7.28{\cdot}10^{-7}$
BEIVKF4	$8.27{\cdot}10^{-4}$	$\pm\ 5.09{\cdot}10^{-5}$	-		$9.59{\cdot}10^{-4}$	$\pm\ 1.86{\cdot}10^{-5}$
BEIVKF5	$2.53{\cdot}10^{-3}$	$\pm\ 1.14{\cdot}10^{-4}$	$2.67{\cdot}10^{-3}$	$\pm\ 2.44{\cdot}10^{-4}$	-	
Filter	mean($\hat{P}_y$)		mean(P_y)		mean($\bar{P}_y$)	
BEIVKF1	$2.03{\cdot}10^{-2}$	$\pm\ 1.57{\cdot}10^{-3}$	$2.05{\cdot}10^{-2}$	$\pm\ 5.58{\cdot}10^{-17}$	-	
BEIVKF2	$7.19{\cdot}10^{-2}$	$\pm\ 9.80{\cdot}10^{-3}$	$7.25{\cdot}10^{-2}$	$\pm\ 3.95{\cdot}10^{-3}$	$2.35{\cdot}10^{-2}$	$\pm\ 3.10{\cdot}10^{-4}$
BEIVKF3	$2.04{\cdot}10^{-2}$	$\pm\ 1.23{\cdot}10^{-3}$	-		$2.03{\cdot}10^{-2}$	$\pm\ 1.82{\cdot}10^{-5}$
BEIVKF4	$2.07{\cdot}10^{-2}$	$\pm\ 1.27{\cdot}10^{-3}$	-		$2.40{\cdot}10^{-2}$	$\pm\ 4.65{\cdot}10^{-4}$
BEIVKF5	$6.32{\cdot}10^{-2}$	$\pm\ 2.86{\cdot}10^{-3}$	$6.68{\cdot}10^{-2}$	$\pm\ 6.11{\cdot}10^{-3}$	-	

Table 6.1: Mean and standard deviation of Monte-Carlo results for all five filters.

filter exhibits the smallest estimation error variance. The variances determined by this filter (P_u and P_y) seem to be in accordance with the sample variances $\hat{P}_u$, $\hat{P}_y$. The pragmatic BEIVKF2 approach appears to perform worst for the given example. As already observed for the state estimation error in Section 6.4.1, the variances of the input and output estimation errors determined by the BEIVKF2, which are $\bar{P}_u$ and $\bar{P}_y$, provide false information. The latter quantities are too optimistic whereas the actual variances, which have been derived in Section 6.4.1, are larger. The actual variances P_u and P_y are in accordance with the sample estimates $\hat{P}_u$, $\hat{P}_y$. The two nonlinear filters BEIVKF3 and BEIVKF4 perform best, both obtain a performance very close to the benchmark filter BEIVKF1. Consequently, these filters can be regarded as quasi-

optimal for the setup considered. In addition, it should be noted here that the 'false' variances $\bar{P}_u$ and $\bar{P}_y$, which are computed by the filters appear to be close to the sample estimates. Note that the actual variances P_u and P_y are not provided for these cases since an error analysis has not been carried out. The BEIVKF5 performance is slightly superior to that of the BEIVKF2, however, it is outperformed by the other filters. Whilst the BEIVKF5 is optimal for state dependent noise systems, it is only suboptimal for the filtering of bilinear EIV systems. This may appear somewhat intuitive by realising that the BEIVKF5 only makes use of the mean and variance of u_{0_k} (cf. Remark 6.4), whereas the optimal filter requires the exact value of the system input. The BEIVKF3, in contrast, utilises the conditional mean estimate of the input which, provided convergence occurs, can yield superior filtering results.

Finally, the performance indices defined in (6.19) are assessed. Their values are stored for each Monte-Carlo iteration and the corresponding mean values and standard deviations are given in Table 6.2. The results appear to be in alignment with

Filter	mean(M_u)		mean(M_y)	
BEIVKF1	91.00	± 0.30	91.00	± 0.30
BEIVKF2	83.19	± 1.04	83.19	± 1.04
BEIVKF3	90.99	± 0.32	90.99	± 0.32
BEIVKF4	90.90	± 0.33	90.90	± 0.33
BEIVKF5	84.14	± 0.47	84.14	± 0.47

Table 6.2: Monte-Carlo mean and standard deviations of the filter performance indices; removed noise in percentage.

those obtained for the variances of the estimation errors: The BEIVKF1 performs best followed by the BEIVKF3 and BEIVKF4. The BEIVKF2 and BEIVKF5 remove the least amount of noise from the input and output signals, where the latter appears to be slightly superior. The standard deviations of the BEIVKF2 are quite large in comparison to the other algorithms. As pointed out in Section 6.4.1, this is likely to be due to the fact that the stability condition of the bilinear system is not satisfied at all times. The fact that the values for M_u and M_y are identical for each filter is not a general property, but is rather a peculiarity of the particular simulation setup.

6.6 Overview & discussion

The different approaches for bilinear EIV filtering are captured in Table 6.3. Further approaches are feasible by coupling the bilinearity with the input matrix $\mathcal{B}$, which allows other suboptimal filters to be derived. The problem formulation is, however, similar to the settings discussed within this chapter. In addition, it might appear tempting to deal with (6.41)-(6.42) as a state dependent noise system. That is, to compute the covariance matrix of the state dependent noise $\bar{v}_k$, whilst considering an LTV system

Filter	System		Model for KF design	
	System matrix	State noise	System matrix	State noise
BEIVKF1	$\mathcal{A}+\mathcal{N}u_{0_k}$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k$	$\mathcal{A}_k^a=\mathcal{A}+\mathcal{N}u_{0_k}$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k$
BEIVKF2	$\mathcal{A}+\mathcal{N}u_{0_k}$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k$	$\mathcal{A}_k^d=\mathcal{A}+\mathcal{N}u_k$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k$
BEIVKF3	$\mathcal{A}+\mathcal{N}u_{0_k}$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k$	$\hat{\mathcal{A}}_k=\mathcal{A}+\mathcal{N}\hat{u}_{0_k}$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k$
BEIVKF4	$\mathcal{A}+\mathcal{N}u_k$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k-\mathcal{N}\tilde{u}_k x_k$	$\mathcal{A}_k^d=\mathcal{A}+\mathcal{N}u_k$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k-\mathcal{N}\tilde{u}_k\hat{x}_{k\|k-1}$
BEIVKF5	$\mathcal{A}$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k+\mathcal{N}x_k u_{0_k}$	$\mathcal{A}$	$\mathcal{G}w_k-\mathcal{B}\tilde{u}_k+\mathcal{N}x_k u_{0_k}$

Table 6.3: Different interpretation of bilinear EIV system and model for KF design used in various filters.

with $\mathcal{A}_k^d=\mathcal{A}+\mathcal{N}u_k$. The covariance matrix $\Sigma_{\bar{v}}^k$ in this case, however, depends on the (unconditional) mean of the state, which, in turn, depends on the unknown input u_{0_k}. This prevents, of course, the optimal filter within a state dependent noise setting to be derived.

A fact which has not been discussed so far, is the choice of admissible input signals. If the bilinear system is interpreted as a LTV system, the system matrix, hence the equivalent poles of the system, are a function of the input u_{0_k}. In order to ensure stability of the system, the input is to be chosen, such that the poles of the corresponding LTV system remain within the unit circle at all times. Therefore, when dealing with bilinear systems, the input is usually assumed to be bounded, in order to ensure stability. Note that this disqualifies the choice of a Gaussian input signal.

The choice of a bounded input also has implications for the distribution of the state, hence for the performance of the filters. If the true input is not a known deterministic signal, this will generally yield a non-Gaussian distribution for the state. Hence, there will be nonlinear filters which achieve superior performance. Depending on the nature of the input (as well as the nature of the noise sequences and the initial state) it might be beneficial to consider nonlinear filtering approaches within a Bayesian framework, to address the bilinear EIV filtering problem. This, as well as the extension towards more general nonlinear system representations, might be an interesting area of further work.

Following the interpretation of the bilinear EIV system as a linear system with state dependent noise, the development of robust bilinear EIV filters within the H_∞ framework also appears to be an area of potential further work (cf. Wang & Balakrishnan 2002, Wang & Qiao 2002).

6.7 Concluding remarks

The errors-in-variables (EIV) filtering problem, i.e. the estimation of noise-free input and output sequences based on noisy measurements has been considered for a class of bilinear systems. By using similar techniques as for the filtering of linear EIV systems and bilinear (non-EIV) systems, it has been shown that the optimal filter, in the minimum variance sense, requires the input signal to be an exactly known deterministic

quantity, hence it is infeasible in the bilinear EIV case. Consequently, attention has been focused on the development of feasible suboptimal filtering approaches. Different Kalman filter design strategies are possible, by interpreting the bilinear EIV system from different viewpoints. These are:

1. Linear time-varying system with uncertain system matrix.
2. Linear time-varying system with state dependent noise.
3. Linear time-invariant system with noise dependent on the state as well as the true input.

Using the first interpretation leads to the development of one linear and one nonlinear suboptimal EIV Kalman filter (EIVKF). The linear filter simply utilises the measured input for the filter design, whilst the nonlinear filter uses the conditional mean estimate of the input. For the linear filter, an error analysis is carried out. This allows for a qualitative assessment of the deterioration of filter performance, and could be utilised to make a decision on whether to apply the filter in practice or not. The second point of view yields a nonlinear filter which can be considered to be a straightforward application of the extended Kalman filter (EKF), which uses the most recent state estimate for the design of the Kalman filter equations. The third interpretation allows the application of a Kalman filter for a system with stochastic parameters, which are also known within the literature as stochastic systems with multiplicative noise. A simulation study, which compares the different filters for a particular scenario, has shown that the nonlinear filters achieve promising performances. Indeed, at least in the particular setup considered, the nonlinear filters are found to exhibit performances which are very close to the that of the benchmark filter and could therefore be considered as being quasi-optimal. The linear filters exhibit an inferior performance in simulation, however, they might be superior in other situations. Whilst further analysis might be necessary, the proposed algorithms for the bilinear EIV filtering problem allow the user to select a filter which is most suitable for a particular problem at hand.

Chapter 7

Errors-in-variables filtering for parameter estimation

Contents

Nomenclature

$\mathcal{A}_0$, $\mathcal{B}_0$, $\mathcal{C}_0$, $\mathcal{D}_0$, $\mathcal{G}$	System matrices
$\mathcal{A}(\theta)$, $\mathcal{B}(\theta)$, $\mathcal{C}(\theta)$, $\mathcal{D}(\theta)$.	Model matrices (general)
$\mathcal{A}_k$, $\mathcal{B}_k$, $\mathcal{C}_k$, $\mathcal{D}_k$	Model matrices (depending on $\hat{\theta}_k$)
e_k	Reformulated output noise
$F(\theta, x, u)$	Jacobian
F_k	Jacobian (depending on $\hat{\theta}_k$)
$H(\theta, x, u)$	Jacobian
H_k	Jacobian (depending on $\hat{\theta}_k$)
$J(\theta, S^{-1}, \varepsilon)$	Jacobian
K_k, $K_k(\theta)$	Kalman gain
$\mathscr{K}_k^{(i)}$	Derivative of $K_k(\theta)$ with respect to θ_i
L_k	Gain for JEKF parameter estimator
M_u, M_y	Filter performance indices for input and output, respectively
$P_{k\|k-1}$, $P_{k\|k-1}(\theta)$	Error covariance matrix of Kalman filter
$\mathscr{P}_k^{(i)}$	Derivative of $P_{k\|k-1}(\theta)$ with respect to θ_i
P_{1_k}, P_{2_k}, P_{3_k}	Covariance matrices of joint Kalman filter for state and parameter estimation
R_k	Approximate Hessian
S_k, $S_k(\theta)$	Innovations covariance matrix
$\mathscr{S}_k^{(i)}$	Derivative of $S_k(\theta)$ with respect to θ_i
$T(\theta)$, T_k	Auxiliary matrix
$u_{0_k}^*$	Intermediate estimate of u_{0_k}
v_k	Reformulated process noise
$V(\theta)$	Prediction error method cost function
$\breve{V}(\theta)$	Cost function corresponding to modified predictor $\breve{y}_k(\theta)$
$V_\epsilon(\theta)$	Cost function corresponding to symmetric innovation $\epsilon_k(\theta)$
w_k	Process noise
W_k, $W_k(\theta)$	Jacobian of $\hat{x}_{k\|k-1}(\theta)$
$\breve{W}_k$, $\breve{W}_k(\theta)$	Jacobian of $\hat{x}_{k\|k-1}(\theta)$ corresponding to modified predictor $\breve{y}_k(\theta)$
x_k	System state
$\hat{x}_{k\|k-1}$, $\hat{x}_{k\|k-1}(\theta)$	System state estimate
$\breve{y}_k$, $\breve{y}_k(\theta)$	Modified predictor
δ_{kl}	Kronecker delta function
ε_k, $\varepsilon_k(\theta)$	Innovation
$\breve{\varepsilon}_k$, $\breve{\varepsilon}_k(\theta)$	Innovation corresponding to modified predictor $\breve{y}_k(\theta)$
ϵ_k, $\epsilon_k(\theta)$	Symmetric innovation
$\eta_k(\theta)$, η_k	Negative gradient of symmetric innovation $\epsilon_k(\theta)$
Λ	Weighting matrix
$\Sigma_e(\theta)$, $\Sigma_v(\theta)$, $\Sigma_{ve}(\theta)$ ...	Noise covariance matrices (general)
Σ_e^k, Σ_v^k, Σ_{ve}^k	Noise covariance matrices (depending on $\hat{\theta}_k$)
ψ_k, $\psi_k(\theta)$	Gradient of predictor

Preliminary reading: Sections 2.2, 2.4.5, 2.5.1, 2.5.2, 2.5.4.

7.1 Introduction

This chapter explores novel avenues to combine errors-in-variables (EIV) filtering, i.e. the estimation of noise-free input and output signals, with EIV system identification techniques. As a starting point, an EIV extended Kalman filter for joint state and parameter estimation (JEKF) is developed, which is able to estimate the system states, the system parameters as well as the noise-free inputs and noise-free outputs of an EIV system. The system parameters obtained, however, appear to be biased in the presence of input measurement noise. This is due to the fact, that the parameter estimator obtained from the JEKF is closely related to the recursive prediction error method (RPEM), which is known to yield biased estimates when applied directly to identify an EIV system. In order to further investigate the JEKF approach, the 'true' RPEM method is derived when applied to an EIV state space system. This, in turn, leads to the proposal for a modification of the JEKF as well as the RPEM approach, in order to reduce the bias within an EIV setup. Finally, a novel identification method for EIV state space systems is developed, by introducing a predictor which accounts for the symmetry of the EIV framework and whose resulting cost function minimises the distance between the noisy inputs and outputs and their filtered counterparts. A recursive implementation based on the standard RPEM technique is also developed and analysed in simulation.

Section 7.2 introduces the setup and assumptions. Section 7.3 derives the JEKF for the EIV case. The development of this section has also been published in (Linden, Vinsonneau & Burnham 2007*c*). Section 7.4 provides a detailed derivation of the RPEM when applied to an EIV system, whilst Section 7.5 derives the novel identification technique based on symmetric innovations. Concluding remarks are given in Section 7.6.

7.2 Preliminaries

Consider the linear time-invariant (LTI) EIV state-space system given by

$$x_{k+1} = \mathcal{A}_0 x_k + \mathcal{B}_0 u_{0_k} + \mathcal{G} w_k, \tag{7.1a}$$

$$y_{0_k} = \mathcal{C}_0 x_k + \mathcal{D}_0 u_{0_k}, \tag{7.1b}$$

$$u_k = u_{0_k} + \tilde{u}_k, \tag{7.1c}$$

$$y_k = y_{0_k} + \tilde{y}_k, \tag{7.1d}$$

where $x_k \in \mathbb{R}^{n_x}$ denotes the state vector of the system, $y_k \in \mathbb{R}^{n_y}$ the measured output, $u_k \in \mathbb{R}^{n_u}$ the measured input and $\mathcal{A}_0$, $\mathcal{B}_0$, $\mathcal{C}_0$, $\mathcal{D}_0$ and $\mathcal{G}$ are matrices of appropriate dimensions[1]. The initial state x_0 is assumed to be a random vector with mean $\bar{x}_0$ and

[1]Without loss of generality, and for convenience only, it is assumed that the matrix $\mathcal{G}$ is known.

covariance matrix P_0. In addition the following assumption is introduced.

AN6b The noise sequences $\tilde{u}_k$, $\tilde{y}_k$ and w_k are assumed to be stationary, zero mean, white, independent of u_{0_k} and are characterised by the known covariance matrices

$$E\left[\begin{bmatrix} x_0 \\ \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix} \begin{bmatrix} x_0^T & \tilde{u}_l^T & \tilde{y}_l^T & w_l \end{bmatrix}\right] = \begin{bmatrix} P_0 & 0 & 0 & 0 \\ 0 & \Sigma_{\tilde{u}} & \Sigma_{\tilde{u}\tilde{y}} & 0 \\ 0 & \Sigma_{\tilde{u}\tilde{y}}^T & \Sigma_{\tilde{y}} & 0 \\ 0 & 0 & 0 & \Sigma_w \end{bmatrix} \delta_{kl}. \tag{7.2}$$

The model corresponding to (7.1) is given by

$$x_{k+1}(\theta) = \mathcal{A}(\theta)x_k(\theta) + \mathcal{B}(\theta)u_{0_k} + \mathcal{G}w_k, \tag{7.3a}$$
$$y_{0_k} = \mathcal{C}(\theta)x_k(\theta) + \mathcal{D}(\theta)u_{0_k}, \tag{7.3b}$$
$$u_k = u_{0_k} + \tilde{u}_k, \tag{7.3c}$$
$$y_k = y_{0_k} + \tilde{y}_k, \tag{7.3d}$$

where θ denotes the parameter vector. Using similar techniques as in Section 2.5.2, (7.3) can be re-expressed as

$$x_{k+1}(\theta) = \mathcal{A}(\theta)x_k(\theta) + \mathcal{B}(\theta)u_k + v_k(\theta), \tag{7.4a}$$
$$y_k = \mathcal{C}(\theta)x_k(\theta) + \mathcal{D}(\theta)u_k + e_k(\theta), \tag{7.4b}$$

where

$$v_k(\theta) = \mathcal{G}w_k - \mathcal{B}(\theta)\tilde{u}_k, \tag{7.5a}$$
$$e_k(\theta) = \tilde{y}_k - \mathcal{D}(\theta)\tilde{u}_k. \tag{7.5b}$$

The noise covariance matrices corresponding to (7.5) are given by

$$\Sigma_v(\theta) = \mathcal{G}\Sigma_w\mathcal{G}^T + \mathcal{B}(\theta)\Sigma_{\tilde{u}}\mathcal{B}^T(\theta), \tag{7.6a}$$
$$\Sigma_e(\theta) = \Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T\mathcal{D}^T(\theta) - \mathcal{D}(\theta)\Sigma_{\tilde{u}\tilde{y}} + \mathcal{D}(\theta)\Sigma_{\tilde{u}}\mathcal{D}^T(\theta), \tag{7.6b}$$
$$\Sigma_{ve}(\theta) = \mathcal{B}(\theta)\Sigma_{\tilde{u}}\mathcal{D}^T(\theta) - \mathcal{B}(\theta)\Sigma_{\tilde{u}\tilde{y}}. \tag{7.6c}$$

Note that the noise sequences, hence their covariance matrices, are dependent on θ.

7.3 Application of the JEKF to the EIV system

7.3.1 Direct application of the JEKF

Using equation (7.4), an application of the JEKF, which has been given in Section 2.5.4, appears to be straightforward, by simply replacing z_k with y_k. The only difference to

the case considered in Section 2.5.4 is that the process and output noise sequences are functions of the unknown parameter vector θ. For the application of the EKF, this is not a burden, since a Taylor expansion of $v_k(\theta)$ and $e_k(\theta)$ around $\hat{\theta}_k$ (cf. p. 194-195 in Anderson & Moore 1979) yields $v_k(\hat{\theta})$ and $e_k(\hat{\theta})$, i.e. θ is simply replaced by its most recent estimate. As a consequence, the corresponding covariance matrices depend on $\hat{\theta}_k$ and are readily given by $\Sigma_v(\hat{\theta}_k)$, $\Sigma_e(\hat{\theta}_k)$ and $\Sigma_{ve}(\hat{\theta}_k)$. After these minor modifications, the JEKF given in Algorithm 2.4 can be directly applied to the EIV state space representation (7.4). The corresponding algorithm, which is denoted EIV-JEKF1, can be summarised as follows (cf. Algorithm 2.4 on page 46).

Algorithm 7.1 (EIV-JEKF1).

$$\varepsilon_k = y_k - \mathcal{C}_k \hat{x}_{k|k-1} - \mathcal{D}_k u_k \tag{7.7a}$$
$$S_k = \mathcal{C}_k P_{1_k} \mathcal{C}_k^T + \mathcal{C}_k P_{2_k} H_k^T + H_k P_{2_k}^T \mathcal{C}_k^T + H_k P_{3_k} H_k^T + \Sigma_e^k \tag{7.7b}$$
$$K_k = [\mathcal{A}_k P_{1_k} \mathcal{C}_k^T + F_k P_{2_k}^T \mathcal{C}_k^T + \mathcal{A}_k P_{2_k} H_k^T + F_k P_{3_k} H_k^T + \Sigma_{ve}^k] S_k^{-1} \tag{7.7c}$$
$$\hat{x}_{k+1|k} = \mathcal{A}_k \hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k \varepsilon_k \tag{7.7d}$$
$$L_k = \left[P_{2_k}^T \mathcal{C}_k^T + P_{3_k} H_k^T \right] S_k^{-1} \tag{7.7e}$$
$$\hat{\theta}_{k+1} = \hat{\theta}_k + L_k \varepsilon_k \tag{7.7f}$$
$$P_{1_{k+1}} = \mathcal{A}_k P_{1_k} \mathcal{A}_k^T + \mathcal{A}_k P_{2_k} F_k^T + F_k P_{2_k}^T \mathcal{A}_k^T + F_k P_{3_k} F_k^T - K_k S_k K_k^T + \Sigma_v^k \tag{7.7g}$$
$$P_{2_{k+1}} = \mathcal{A}_k P_{2_k} + F_k P_{3_k} - K_k S_k L_k^T \tag{7.7h}$$
$$P_{3_{k+1}} = P_{3_k} - L_k S_k L_k^T + \Sigma_d \tag{7.7i}$$
$$\hat{y}_{0_k} = y_k - \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T \mathcal{D}_k^T \right] S_k^{-1} \varepsilon_k \tag{7.7j}$$
$$\hat{u}_{0_k} = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}} \mathcal{D}_k^T \right] S_k^{-1} \varepsilon_k \tag{7.7k}$$

As before, the Jacobians are obtained via

$$F_k = F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) \in \mathbb{R}^{n_x \times n_\theta}, \tag{7.8a}$$
$$H_k = H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) \in \mathbb{R}^{n_y \times n_\theta}, \tag{7.8b}$$

where (cf. (2.119))

$$F(\hat{\theta}, x, u) = \frac{\partial}{\partial \theta} \left[\mathcal{A}(\theta) x + \mathcal{B}(\theta) u \right] \big|_{\theta=\hat{\theta}}, \tag{7.9a}$$
$$H(\hat{\theta}, x, u) = \frac{\partial}{\partial \theta} \left[\mathcal{C}(\theta) x + \mathcal{D}(\theta) u \right] \big|_{\theta=\hat{\theta}}. \tag{7.9b}$$

In addition, the convenient notation

$$\mathcal{A}_k \triangleq \mathcal{A}(\hat{\theta}_k), \qquad \mathcal{B}_k \triangleq \mathcal{B}(\hat{\theta}_k), \qquad \mathcal{C}_k \triangleq \mathcal{C}(\hat{\theta}_k), \qquad \mathcal{D}_k \triangleq \mathcal{D}(\hat{\theta}_k), \tag{7.10}$$

and

$$\Sigma_v^k \triangleq \Sigma_v(\hat{\theta}_k), \qquad \Sigma_e^k \triangleq \Sigma_e(\hat{\theta}_k), \qquad \Sigma_{ve}^k \triangleq \Sigma_{ve}(\hat{\theta}_k), \qquad \hat{x}_{k|k-1} \triangleq \hat{x}_{k|k-1}(\hat{\theta}_k), \tag{7.11}$$

has been used. Note that the innovations covariance matrix is given by S_k in the JEKF case (whilst it was denoted Σ_ε^k given by (2.106c) for the linear case). As outlined in Section 2.5.4, variations in the parameters to be estimated can be dealt with by specifying a corresponding error covariance matrix Σ_d, which is added to P_{3_k}. The only novelty in Algorithm 7.1 with respect to the non-EIV version is the additional estimation of $\hat{y}_{0_k}$ and $\hat{u}_{0_k}$ for which the error covariance matrices are given by (cf. (2.108))

$$P_u^k = \Sigma_{\tilde{u}} - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}}\mathcal{D}_k^T\right] S_k^{-1} \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}}\mathcal{D}_k^T\right]^T, \tag{7.12a}$$

$$P_y^k = \Sigma_{\tilde{y}} - \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}^T\mathcal{D}_k^T\right] S_k^{-1} \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T\mathcal{D}_k^T\right]^T. \tag{7.12b}$$

It has been shown that an EKF for joint state and parameter estimation can be obtained in a straightforward manner when applied to an EIV state space system. However, there is one 'hitch' with the previously derived algorithm. Recall that the JEKF consists effectively of a state estimator which is cross-coupled with a parameter estimator, whose structure is inspired by the EKF. The cross-coupling means that the state estimator uses the most recent parameter estimates provided by the parameter estimator, which, in turn uses the most recent state estimate to arrive at the updated parameter estimates. As pointed out in (Ljung 1979), the EKF parameter estimator (cf. Section 2.5.4) is closely related to a recursive prediction error method (RPEM) assuming a constant model. However, it is pointed out in (Söderström 1981), that a direct application of the prediction error method (PEM) to an EIV problem does not yield consistent estimates, since the corresponding cost function is not minimised for the true parameters if the presence of the input noise is neglected. This leads to the following proposition.

Conjecture 7.1. The parameter estimates of the EIV-JEKF1 algorithm are biased.

A consequence of this conjecture is that, due to the cross-coupling, the performance of the state estimator will suffer as well, since, even in the asymptotic case, a systematic mismatch between the system and the model used for filtering will remain. Since the input and output estimates are based on the state estimates, the quality of the former will be affected, too. To avoid any ambiguities, the bias mentioned in Proposition 7.1 is not assumed to be due to the missing cross-coupling term $[\partial K(\theta)/\partial\theta]\varepsilon_k$, which can be a source of bias and divergence in the non-EIV case (cf. Section 2.5.4), but due to the EIV nature of the problem. This implies that Proposition 7.1 still holds when an innovations representation would be utilised to derive the EIV-JEKF1 equations, which ensures consistency in a non-EIV setting. In order to substantiate Proposition

7.1, the following example is considered.

7.3.2 Numerical example

Consider an EIV LTI single-input single-output (SISO) dynamical system which is defined by

$$y_{0_k} = -a_1 y_{0_{k-1}} - a_2 y_{0_{k-2}} + b_0 u_{0_k} + b_1 u_{0_{k-1}}, \tag{7.13a}$$
$$u_k = u_{0_k} + \tilde{u}_k, \tag{7.13b}$$
$$y_k = y_{0_k} + \tilde{y}_k, \tag{7.13c}$$

with parameter vector[2]

$$\theta = \begin{bmatrix} a_1 & a_2 & b_0 & b_1 \end{bmatrix}^T = \begin{bmatrix} -0.3 & 0.2 & -4.5 & 5.4 \end{bmatrix}^T. \tag{7.14}$$

A possible state space realisation is given by

$$\mathcal{A}(\theta) = \begin{bmatrix} 0 & 1 \\ -a_2 & -a_1 \end{bmatrix}, \qquad \mathcal{B}(\theta) = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$
$$\mathcal{C}(\theta) = \begin{bmatrix} -a_2 b_0 & b_1 - a_1 b_0 \end{bmatrix}, \qquad \mathcal{D}(\theta) = b_0. \tag{7.15}$$

The covariance matrix of the input and output measurement noise sequences is given by

$$E\left[\begin{bmatrix} \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix} \begin{bmatrix} \tilde{u}_l & \tilde{y}_l & w_l \end{bmatrix}\right] = \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.8 & 5 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \delta_{kl}, \tag{7.16}$$

which corresponds to a signal-to-noise ratio of 15dB and 21dB for the input and output, respectively. The matrix $\mathcal{G}$ is set to $\mathcal{G} = [1 \;\; 0]^T$, which means that only the first element of the state is affected by process noise. The input is chosen to be a Gaussian zero mean random sequence of unity variance. The Jacobians are given by

$$\begin{aligned} F_k &= F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) \\ &= \frac{\partial}{\partial \theta} \left[\mathcal{A}(\theta)\hat{x}_{k|k-1} + \mathcal{B}(\theta)u_k\right] \big|_{\theta=\hat{\theta}} \\ &= \frac{\partial}{\partial \theta} \left[\begin{matrix} \hat{x}^2_{k|k-1} \\ -a_2\hat{x}^1_{k|k-1} - a_1\hat{x}^2_{k|k-1} + u_k \end{matrix}\right]\Bigg|_{\theta=\hat{\theta}_k} \\ &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\hat{x}^2_{k|k-1} & -\hat{x}^1_{k|k-1} & 0 & 0 \end{bmatrix}, \end{aligned} \tag{7.17}$$

[2]This is an arbitrarily chosen non-minimum phase system. It is actually the system studied in Chapter 6 with the bilinear term omitted.

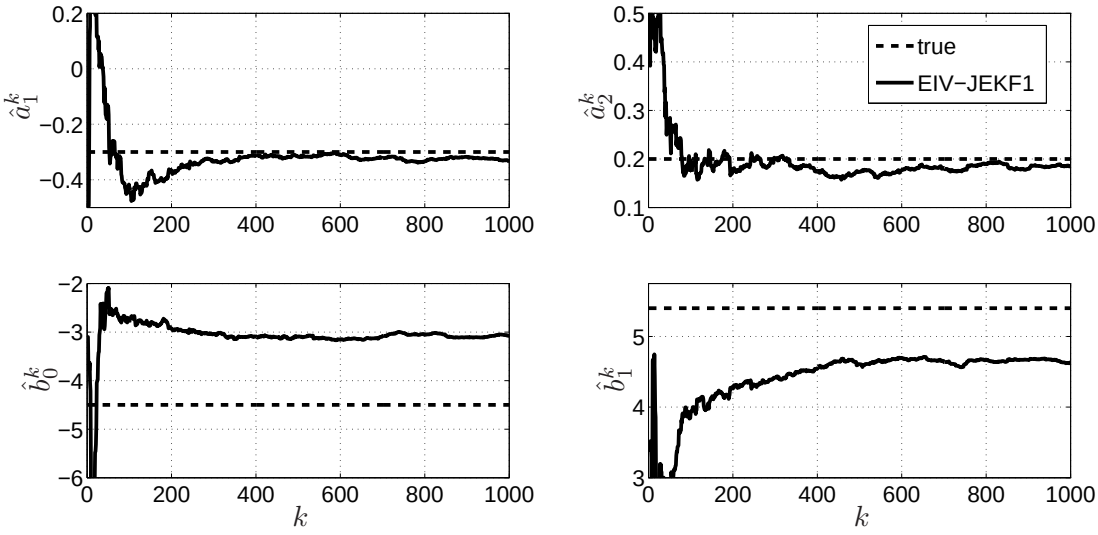


Figure 7.1: Parameter estimates obtained by the EIV-JEKF1 algorithm.

and

$$\begin{aligned} H_k &= H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) \\ &= \frac{\partial}{\partial\theta}\left[\mathcal{C}(\theta)\hat{x}_{k|k-1} + \mathcal{D}(\theta)u_k\right]\big|_{\theta=\hat{\theta}} \\ &= \frac{\partial}{\partial\theta}\left(-a_2 b_0 \hat{x}^1_{k|k-1} + b_1\hat{x}^2_{k|k-1} - a_1 b_0 \hat{x}^2_{k|k-1} + b_0 u_k\right)\Big|_{\theta=\hat{\theta}_k} \\ &= \left[-b_0^k\hat{x}^2_{k|k-1}, \quad -b_0^k\hat{x}^1_{k|k-1}, \quad (-a_2^k\hat{x}^1_{k|k-1} - a_1^k\hat{x}^2_{k|k-1} + u_k), \quad \hat{x}^2_{k|k-1}\right], \end{aligned} \tag{7.18}$$

where $\hat{x}^n_{k|k-1}$ denotes the nth state estimate. The EIV-JEKF1 algorithm is utilised to estimate the states, the parameters as well as the noise-free input and noise-free output of the system. The filter is initialised with

$$\hat{\theta}_0 = \begin{bmatrix} -0.5 & 0.4 & -3.1 & 3.4 \end{bmatrix}^T, \qquad P_0 = 100I, \qquad x_0 = 0. \tag{7.19}$$

In addition, a projection facility is utilised, in order to ensure that all eigenvalues of $\mathcal{A}_k - K_k\mathcal{C}_k$ lie strictly within the unit circle (see Section 2.5.4). The parameter estimates obtained by the EIV-JEKF1 algorithm for $N = 1000$ samples are shown in Figure 7.1. It is observed that whilst a_1 and a_2 appear to be estimated without (or with very little) bias, the parameters b_0 and b_1 are notably biased. Although this single realisation can only be considered to be of an indicative character, it seems to underpin Proposition 7.1.

For completeness, consider the filter performance with regard to obtaining filtered

input and output signals, given by (6.19)

$$M_u = 100\frac{||u_0 - u||_2 - ||u_0 - \hat{u}_0||_2}{||u_0 - u||_2}, \tag{7.20a}$$

$$M_y = 100\frac{||y_0 - y||_2 - ||y_0 - \hat{y}_0||_2}{||y_0 - y||_2}, \tag{7.20b}$$

which can be interpreted as the amount of noise, expressed as a percentage, which is removed, respectively, from the noisy input and output signals. Note that the variables without time index (u, y, etc. in (7.20)) denote the corresponding sequences from 1 to N. For this particular example, the values for M_y and M_y are found to be 41% and 49%, respectively. Consequently, the filter is able to filter the input and output of the EIV system, although the estimated model parameters, which are utilised to arrive at the filtered input and output signals, are biased.

7.3.3 Modified JEKF design

The previous example indicates that the parameter estimator of the EIV-JEKF1 algorithm yields biased estimates. If the true input was available, it would be possible to design a modified parameter estimator based on the EIV-JEKF1, such that the bias in the estimates is reduced. Whilst an explanation and further discussion is postponed until Section 7.4, this conjecture is captured in the following proposition.

Proposition 7.1. If the Jacobian H_k (cf. (7.8b)) within the EIV-JEKF1 algorithm is replaced by $H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_{0_k})$, where the noise-free input is utilised rather than its noisy measurement, the accordingly modified EIV-JEKF algorithm is able to provide estimates with reduced bias.

Although the difference with respect to Algorithm 7.1 is trivial, the algorithm, which is denoted EIV-JEKF2, is, for completeness, summarised as follows.

Algorithm 7.2 (EIV-JEKF2).

$$\varepsilon_k = y_k - \mathcal{C}_k\hat{x}_{k|k-1} - \mathcal{D}_k u_k \tag{7.21a}$$

$$H_k = H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_{0_k}) \tag{7.21b}$$

$$S_k = \mathcal{C}_k P_{1_k}\mathcal{C}_k^T + \mathcal{C}_k P_{2_k}H_k^T + H_k P_{2_k}^T\mathcal{C}_k^T + H_k P_{3_k}H_k^T + \Sigma_e^k \tag{7.21c}$$

$$K_k = [\mathcal{A}_k P_{1_k}\mathcal{C}_k^T + F_k P_{2_k}^T\mathcal{C}_k^T + \mathcal{A}_k P_{2_k}H_k^T + F_k P_{3_k}H_k^T + \Sigma_{ve}^k]S_k^{-1} \tag{7.21d}$$

$$\hat{x}_{k+1|k} = \mathcal{A}_k\hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k\varepsilon_k \tag{7.21e}$$

$$L_k = \left[P_{2_k}^T\mathcal{C}_k^T + P_{3_k}H_k^T\right]S_k^{-1} \tag{7.21f}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + L_k\varepsilon_k \tag{7.21g}$$

$$P_{1_{k+1}} = \mathcal{A}_k P_{1_k}\mathcal{A}_k^T + \mathcal{A}_k P_{2_k}F_k^T + F_k P_{2_k}^T\mathcal{A}_k^T + F_k P_{3_k}F_k^T - K_k S_k K_k^T + \Sigma_v^k \tag{7.21h}$$

$$P_{2_{k+1}} = \mathcal{A}_k P_{2_k} + F_k P_{3_k} - K_k S_k L_k^T \tag{7.21i}$$
$$P_{3_{k+1}} = P_{3_k} - L_k S_k L_k^T + \Sigma_d \tag{7.21j}$$
$$\hat{y}_{0_k} = y_k - \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T \mathcal{D}_k^T\right] S_k^{-1} \varepsilon_k \tag{7.21k}$$
$$\hat{u}_{0_k} = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}} \mathcal{D}_k^T\right] S_k^{-1} \varepsilon_k \tag{7.21l}$$

Note that F_k still uses the noisy input u_k, as defined in (7.8a). Although not feasible in practice, this algorithm is considered here as a benchmark for comparison purposes in order to evaluate the subsequently developed algorithms. Furthermore, since the unknown input is estimated within the EIV-JEKF, it appears to be a most natural choice to use this estimate rather than the impractical u_{0_k} within the linearised model, i.e. to cross-couple the input estimation with the parameter estimation. This implies the usage of the Jacobians computed via

$$F_k = F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k), \tag{7.22a}$$
$$H_k = H(\hat{\theta}_k, \hat{x}_{k|k-1}, \hat{u}_{0_k}), \tag{7.22b}$$

where u_{0_k} is approximated via $\hat{u}_{0_k}$ for the determination of the Jacobian H_k. Recall that the input estimate is given by

$$\hat{u}_{0_k} = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}} \mathcal{D}_k^T\right] S_k^{-1} \varepsilon_k. \tag{7.23}$$

Whilst $\mathcal{D}_k$ and ε_k can be computed before the Jacobians are evaluated, the innovations covariance S_k depends on H_k (see (7.21c)). It seems natural, however, to compute an intermediate estimate of u_{0_k} by making use of the approximation $S_k \approx S_{k-1}$. The corresponding estimate is denoted $\hat{u}^*_{0_k}$ and the accordingly modified filter, which is denoted EIV-JEKF3, can be summarised as follows.

Algorithm 7.3 (EIV-JEKF3).

$$\varepsilon_k = y_k - \mathcal{C}_k \hat{x}_{k|k-1} - \mathcal{D}_k u_k \tag{7.24a}$$
$$\hat{u}^*_{0_k} = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}} \mathcal{D}_k^T\right] S_{k-1}^{-1} \varepsilon_k \tag{7.24b}$$
$$H_k = H(\hat{\theta}_k, \hat{x}_{k|k-1}, \hat{u}^*_{0_k}) \tag{7.24c}$$
$$S_k = \mathcal{C}_k P_{1_k} \mathcal{C}_k^T + \mathcal{C}_k P_{2_k} H_k^T + H_k P_{2_k}^T \mathcal{C}_k^T + H_k P_{3_k} H_k^T + \Sigma_e^k \tag{7.24d}$$
$$K_k = [\mathcal{A}_k P_{1_k} \mathcal{C}_k^T + F_k P_{2_k}^T \mathcal{C}_k^T + \mathcal{A}_k P_{2_k} H_k^T + F_k P_{3_k} H_k^T + \Sigma_{ve}^k] S_k^{-1} \tag{7.24e}$$
$$\hat{x}_{k+1|k} = \mathcal{A}_k \hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k \varepsilon_k \tag{7.24f}$$
$$L_k = \left[P_{2_k}^T \mathcal{C}_k^T + P_{3_k} H_k^T\right] S_k^{-1} \tag{7.24g}$$

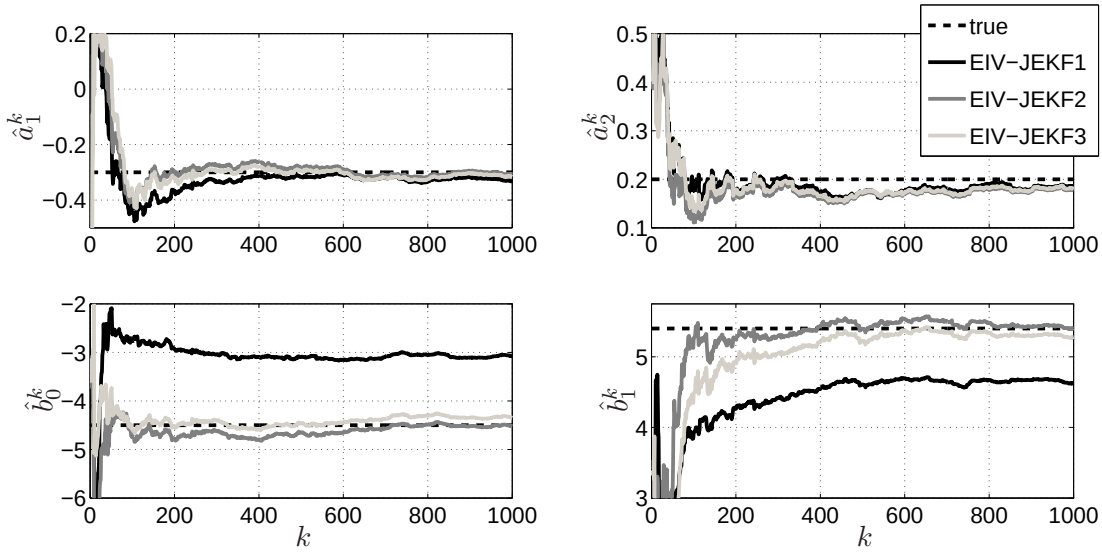


Figure 7.2: Parameter estimates obtained by the EIV-JEKF1, EIV-JEKF2 and EIV-JEKF3 algorithms.

$$\hat{\theta}_{k+1} = \hat{\theta}_k + L_k\varepsilon_k \tag{7.24h}$$

$$P_{1_{k+1}} = \mathcal{A}_k P_{1_k}\mathcal{A}_k^T + \mathcal{A}_k P_{2_k}F_k^T + F_k P_{2_k}^T\mathcal{A}_k^T + F_k P_{3_k}F_k^T - K_k S_k K_k^T + \Sigma_v^k \tag{7.24i}$$

$$P_{2_{k+1}} = \mathcal{A}_k P_{2_k} + F_k P_{3_k} - K_k S_k L_k^T \tag{7.24j}$$

$$P_{3_{k+1}} = P_{3_k} - L_k S_k L_k^T + \Sigma_d \tag{7.24k}$$

$$\hat{y}_{0_k} = y_k - \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T\mathcal{D}_k^T\right] S_k^{-1}\varepsilon_k \tag{7.24l}$$

$$\hat{u}_{0_k} = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}}\mathcal{D}_k^T\right] S_k^{-1}\varepsilon_k \tag{7.24m}$$

In order to substantiate Proposition 7.1 and in order to compare the EIV-JEKF2-3 algorithms with the previously proposed EIV-JEKF1 algorithm, the following example is considered.

7.3.4 Numerical example

Consider an identical setup as in Section 7.3.2. The experiment is repeated and the three EIV-JEKF filters are applied to estimate the states, the parameters and the noise-free input and noise-free output of the system. The parameter estimates obtained by the three filters are shown in Figure 7.2. It is observed that the values of the estimates for a_1 and a_2 are very similar and close to the true values for all three filters. For the estimates of b_0 and b_1, however, it is observed that the values obtained by the EIV-JEKF2 seem to compensate for the bias, which is obtained when the EIV-JEKF1 is applied. This appears to underpin the statement given in Proposition 7.1. In addition, if u_{0_k} is approximated by $\hat{u}_{0_k}^*$ for the design of the Jacobian H_k as it is realised within the

EIV-JEKF3, the bias appears to be reduced as well. However, perhaps not surprisingly, the results seem to be slightly inferior to those obtained by the EIV-JEKF2. This might be an expected result due to the inevitable estimation error of the noise-free input.

So far, no justification has been given, as to why the utilisation of the true input for the design of H_k should result in a reduction of the bias within the EIV-JEKF algorithm. This is addressed in the following section by analysing the 'true' RPEM when applied to the EIV state space system.

7.4 RPEM method for EIV identification

In order to provide a thorough analysis of the JEKF algorithms in the EIV framework, it is necessary to derive the corresponding RPEM. This will give the necessary insight to understand what role the choice of the true input signal plays within the Jacobian H_k and lays the foundation for the development of novel RPEM based methods for EIV system identification. This section derives the RPEM method for a general EIV state space representation which is, apart from minor adjustments, analogous to that presented in (Ljung & Söderström 1983, Appendix 3.B).

7.4.1 Derivation of the RPEM for the EIV case

Consider the EIV state space model given by (7.4)-(7.5)

$$x_{k+1}(\theta) = \mathcal{A}(\theta)x_k(\theta) + \mathcal{B}(\theta)u_k + v_k(\theta), \tag{7.25a}$$
$$y_k = \mathcal{C}(\theta)x_k(\theta) + \mathcal{D}(\theta)u_k + e_k(\theta), \tag{7.25b}$$
$$v_k(\theta) = w_k - \mathcal{B}(\theta)\tilde{u}_k, \tag{7.25c}$$
$$e_k(\theta) = \tilde{y}_k - \mathcal{D}(\theta)\tilde{u}_k. \tag{7.25d}$$

The minimum variance estimates of the state, noise-free input and noise-free output is given by the EIVKF given in Algorithm 2.3, where Σ_ε^k is replaced by $S_k(\theta)$

$$\hat{x}_{k+1|k}(\theta) = \mathcal{A}(\theta)\hat{x}_{k|k-1}(\theta) + \mathcal{B}(\theta)u_k + K_k(\theta)\varepsilon_k(\theta), \tag{7.26a}$$
$$\varepsilon_k(\theta) = y_k - \mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) - \mathcal{D}(\theta)u_k, \tag{7.26b}$$
$$S_k(\theta) = \mathcal{C}(\theta)P_{k|k-1}(\theta)\mathcal{C}^T(\theta) + \Sigma_e(\theta), \tag{7.26c}$$
$$K_k(\theta) = \left[\mathcal{A}(\theta)P_{k|k-1}(\theta)\mathcal{C}^T(\theta) + \Sigma_{ve}(\theta)\right] S_k^{-1}(\theta), \tag{7.26d}$$
$$P_{k+1|k}(\theta) = \mathcal{A}(\theta)P_{k|k-1}(\theta)\mathcal{A}^T(\theta) + \Sigma_v(\theta) - K_k(\theta)S_k(\theta)K_k^T(\theta), \tag{7.26e}$$
$$\hat{y}_{0_k}(\theta) = y_k - \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T\mathcal{D}^T(\theta)\right] S_k^{-1}(\theta)\varepsilon_k(\theta), \tag{7.26f}$$
$$\hat{u}_{0_k}(\theta) = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}}\mathcal{D}^T(\theta)\right] S_k^{-1}(\theta)\varepsilon_k(\theta). \tag{7.26g}$$

The application of the standard PEM usually aims to minimise a cost function which is quadratic in the innovation $\varepsilon_k(\theta)$, i.e.

$$V(\theta) = \frac{1}{2}\varepsilon_k^T(\theta)\Lambda^{-1}\varepsilon_k(\theta), \tag{7.27}$$

where Λ^{-1} is a general weighting matrix, whose optimal choice is given by the innovations covariance matrix.

In order to minimise $V(\theta)$, its gradient with respect to θ is required. Therefore, define

$$\psi_k^T(\theta) \triangleq \frac{d}{d\theta}\left[\mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) + \mathcal{D}(\theta)u_k\right] \in \mathbb{R}^{n_y \times n_\theta}, \tag{7.28}$$

$$W_k(\theta) \triangleq \frac{d}{d\theta}\hat{x}_{k|k-1}(\theta) \in \mathbb{R}^{n_x \times n_\theta}, \tag{7.29}$$

as being the gradient of the one-step-ahead prediction of y_k and the gradient of the state estimate, respectively. Note that (7.28) contains the derivatives of a matrix with respect to a vector $(dC(\theta)/d\theta)$. Such a quantity will be a tensor, i.e. having three indices (the first two indices give the row and the column of $C(\theta)$, whilst the third index corresponds to the ith element of θ). As proposed in (Ljung & Söderström 1983), the following matrices are introduced to avoid such a complex notation. Therefore define

$$F(\hat{\theta}, x, u) \triangleq \frac{\partial}{\partial\theta}\left[\mathcal{A}(\theta)x + \mathcal{B}(\theta)u\right]\big|_{\theta=\hat{\theta}} \in \mathbb{R}^{n_x \times n_\theta}, \tag{7.30a}$$

$$H(\hat{\theta}, x, u) \triangleq \frac{\partial}{\partial\theta}\left[\mathcal{C}(\theta)x + \mathcal{D}(\theta)u\right]\big|_{\theta=\hat{\theta}} \in \mathbb{R}^{n_y \times n_\theta}. \tag{7.30b}$$

This means that $F(\hat{\theta}, x, u)$ is a matrix where the ith column is given by

$$\frac{\partial}{\partial\theta_i}\left[\mathcal{A}(\theta)\right]x + \frac{\partial}{\partial\theta_i}\left[\mathcal{B}(\theta)\right]u. \tag{7.31}$$

This holds analogously for $H(\hat{\theta}, x, u)$. By applying the product rule and using (7.29) as well as (7.30b), but replacing $d/d(\theta)$ by $\partial/\partial(\theta)$ implying partial differentiation, Equation (7.28) becomes

$$\begin{aligned}\psi_k^T(\theta) &= \frac{d}{d\theta}\left[\mathcal{C}(\theta)\right]\hat{x}_{k|k-1}(\theta) + \mathcal{C}(\theta)\frac{d}{d\theta}\left[\hat{x}_{k|k-1}(\theta)\right] + \frac{d}{d\theta}\left[\mathcal{D}(\theta)\right]u_k \\ &= \mathcal{C}(\theta)W_k(\theta) + H(\theta, \hat{x}_{k|k-1}(\theta), u_k).\end{aligned} \tag{7.32}$$

Also note that the gradient of the innovation is given by

$$\begin{aligned}\frac{d}{d\theta}\varepsilon_k(\theta) &= \frac{d}{d\theta}\left[y_k - \mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) - \mathcal{D}(\theta)u_k\right] \\ &= -\psi_k^T(\theta).\end{aligned} \tag{7.33}$$

The immediate objective is now to derive a recursive expression for $W_k(\theta)$, which is

obtained by differentiating (7.26a). This yields

$$W_{k+1}(\theta) = \frac{d}{d\theta}\left[\mathcal{A}(\theta)\right]\hat{x}_{k|k-1}(\theta) + \mathcal{A}(\theta)W_k(\theta) + \frac{d}{d\theta}\left[\mathcal{B}(\theta)\right]u_k + \frac{d}{d\theta}\left[K_k(\theta)\right]\varepsilon_k(\theta) + K_k(\theta)\frac{d}{d\theta}\left[\varepsilon_k(\theta)\right], \tag{7.34}$$

and by making use of (7.30a), (7.30b), (7.32) and (7.33), it holds

$$W_{k+1}(\theta) = \left[\mathcal{A}(\theta) - K_k(\theta)\mathcal{C}(\theta)\right]W_k(\theta) + F(\theta, \hat{x}_{k|k-1}(\theta), u_k) + \frac{d}{d\theta}\left[K_k(\theta)\right]\varepsilon_k(\theta) - K_k(\theta)H(\theta, \hat{x}_{k|k-1}(\theta), u_k). \tag{7.35}$$

In order to obtain the gradient of $V(\theta)$, it remains now to find an expression of $dK(\theta)/d(\theta)$ within (7.35), which is obtained by differentiating (7.26c)-(7.26e). Therefore, introduce the notation

$$\mathscr{K}_k^{(i)} \triangleq \frac{d}{d\theta_i}K_k(\theta), \tag{7.36a}$$

$$\mathscr{S}_k^{(i)} \triangleq \frac{d}{d\theta_i}S_k(\theta), \tag{7.36b}$$

$$\mathscr{P}_k^{(i)} \triangleq \frac{d}{d\theta_i}P_{k|k-1}(\theta). \tag{7.36c}$$

Also introduce P_k, S_k and K_k which are obtained via (7.26c)-(7.26e) with θ being replaced by $\hat{\theta}_k$. Taking the derivative of (7.26c)-(7.26e) with respect to each element of θ individually and evaluating it at the most recent estimate $\hat{\theta}_k$, yields

$$\mathscr{K}_k^{(i)} = \left[\frac{\partial}{\partial\theta_i}\left[\mathcal{A}(\theta)\right]P_{k|k-1}\mathcal{C}_k^T + \mathcal{A}_k\mathscr{P}_k^{(i)}\mathcal{C}_k^T + \mathcal{A}_k P_{k|k-1}\frac{\partial}{\partial\theta_i}\left[\mathcal{C}^T(\theta)\right] + \frac{\partial}{\partial\theta_i}\left[\Sigma_{ve}(\theta)\right]\right]\Bigg|_{\theta=\hat{\theta}_k} S_k^{-1} - K_k\mathscr{S}_k^{(i)}S_k^{-1}, \tag{7.37}$$

$$\mathscr{S}_k^{(i)} = \left[\frac{\partial}{\partial\theta_i}\left[\mathcal{C}(\theta)\right]P_{k|k-1}\mathcal{C}_k^T + \mathcal{C}_k\mathscr{P}_k^{(i)}\mathcal{C}_k^T + \mathcal{C}_k P_{k|k-1}\frac{\partial}{\partial\theta_i}\left[\mathcal{C}^T(\theta)\right] + \frac{\partial}{\partial\theta_i}\left[\Sigma_e(\theta)\right]\right]\Bigg|_{\theta=\hat{\theta}_k}, \tag{7.38}$$

and

$$\mathscr{P}_{k+1}^{(i)} = \left[\frac{\partial}{\partial\theta_i}\left[\mathcal{A}(\theta)\right]P_{k|k-1}\mathcal{A}_k^T + \mathcal{A}_k\mathscr{P}_k^{(i)}\mathcal{A}_k^T + \mathcal{A}_k P_{k|k-1}\frac{\partial}{\partial\theta_i}\left[\mathcal{A}^T(\theta)\right] + \frac{\partial}{\partial\theta_i}\left[\Sigma_v(\theta)\right] - \mathscr{K}_k^{(i)}S_kK_k^T - K_k\mathscr{S}_k^{(i)}K_k^T - K_kS_k\left[\mathscr{K}_k^{(i)}\right]^T\right]\Bigg|_{\theta=\hat{\theta}_k}. \tag{7.39}$$

Note that the convenient notation (7.10) has been used within (7.37)-(7.39). Recall

that the error covariance matrices for the EIV case are given by

$$\Sigma_v(\theta) = \mathcal{G}\Sigma_w\mathcal{G}^T + \mathcal{B}(\theta)\Sigma_{\tilde{u}}\mathcal{B}^T(\theta), \tag{7.40a}$$

$$\Sigma_e(\theta) = \Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T\mathcal{D}^T(\theta) - \mathcal{D}(\theta)\Sigma_{\tilde{u}\tilde{y}} + \mathcal{D}(\theta)\Sigma_{\tilde{u}}\mathcal{D}^T(\theta), \tag{7.40b}$$

$$\Sigma_{ve}(\theta) = \mathcal{B}(\theta)\Sigma_{\tilde{u}}\mathcal{D}^T(\theta) - \mathcal{B}(\theta)\Sigma_{\tilde{u}\tilde{y}}. \tag{7.40c}$$

Their derivatives with respect to θ_i, which are required within (7.37)-(7.39), are therefore given by

$$\frac{\partial}{\partial\theta_i}\Sigma_{v(\theta)} = \frac{\partial}{\partial\theta_i}\left[\mathcal{B}(\theta)\right]\Sigma_{\tilde{u}}\mathcal{B}_k^T + \mathcal{B}_k\Sigma_{\tilde{u}}\frac{\partial}{\partial\theta_i}\left[\mathcal{B}^T(\theta)\right], \tag{7.41a}$$

$$\begin{aligned}\frac{\partial}{\partial\theta_i}\Sigma_e(\theta) = &-\Sigma_{\tilde{u}\tilde{y}}{}^T\frac{\partial}{\partial\theta_i}\left[\mathcal{D}^T(\theta)\right] - \frac{\partial}{\partial\theta_i}\left[\mathcal{D}(\theta)\right]\Sigma_{\tilde{u}\tilde{y}} \\ &+ \frac{\partial}{\partial\theta_i}\left[\mathcal{D}(\theta)\right]\Sigma_{\tilde{u}}\mathcal{D}_k^T + \mathcal{D}_k\Sigma_{\tilde{u}}\frac{\partial}{\partial\theta_i}\left[\mathcal{D}^T(\theta)\right],\end{aligned} \tag{7.41b}$$

$$\frac{\partial}{\partial\theta_i}\Sigma_{ve}(\theta) = \frac{\partial}{\partial\theta_i}\left[\mathcal{B}(\theta)\right]\Sigma_{\tilde{u}}\mathcal{D}_k^T + \mathcal{B}_k\Sigma_{\tilde{u}}\frac{\partial}{\partial\theta_i}\left[\mathcal{D}^T(\theta)\right] - \frac{\partial}{\partial\theta_i}\left[\mathcal{B}(\theta)\right]\Sigma_{\tilde{u}\tilde{y}}. \tag{7.41c}$$

Finally, by setting $dK(\theta)/d\theta$ in (7.35) to $\mathscr{K}_k$, whose ith column is given by $\mathscr{K}_k^{(i)}$, it is possible to compute the gradient of $V(\theta)$ by

$$\frac{d}{d\theta}V(\theta) = -\psi_k(\theta)\Lambda^{-1}\varepsilon_k(\theta), \tag{7.42}$$

where $\psi_k(\theta)$ can now be computed in a recursive way. The minimisation of $V(\theta)$ can be carried out via an iterative Newton method, where it is iterated once as new data arrives. This gives the recursive scheme

$$\hat{\theta}_k = \hat{\theta}_{k-1} + \gamma_k R_k^{-1}\psi_k(\theta)\Lambda^{-1}\varepsilon_k(\theta), \tag{7.43}$$

where R is the Hessian (second derivative of $V(\theta)$). If the latter is approximated via $\psi_k(\theta)\Lambda^{-1}\psi_k^T(\theta)$ for which a recursive expression is obtained in a straightforward manner, a Gauss-Newton algorithm is obtained. The scalar γ_k is a normalising gain, which is set to $1/k$ in the case of no adaptivity, or which becomes $1-\lambda$ in the case of exponential data weighting, where λ denotes the forgetting factor.

7.4.2 Direct application of the RPEM

If the weighting matrix Λ is chosen as the identity matrix, a direct application of the RPEM to the EIV state space system in general form can be summarised as follows.

Algorithm 7.4 (EIV-RPEM1).

$$\varepsilon_k = y_k - \mathcal{C}_{k-1}\hat{x}_{k|k-1} - \mathcal{D}_{k-1}u_k \tag{7.44a}$$

$$\psi_k = W_k^T \mathcal{C}_{k-1}^T + H^T(\hat{\theta}_{k-1}, \hat{x}_{k|k-1}, u_k) \tag{7.44b}$$
$$R_k = R_{k-1} + \gamma_k \left[\psi_k \psi_k^T - R_{k-1}\right] \tag{7.44c}$$
$$\hat{\theta}_k = \hat{\theta}_{k-1} + \gamma_k R_k^{-1} \psi_k \varepsilon_k \tag{7.44d}$$
$$S_k = \mathcal{C}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_e^k \tag{7.44e}$$
$$K_k = \left[\mathcal{A}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_{ve}^k\right] S_k^{-1} \tag{7.44f}$$
$$\hat{x}_{k+1|k} = \mathcal{A}_k \hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k \varepsilon_k \tag{7.44g}$$
$$P_{k+1|k} = \mathcal{A}_k P_{k|k-1} \mathcal{A}_k^T + \Sigma_v^k - K_k S_k K_k^T \tag{7.44h}$$
$$W_{k+1} = \left[\mathcal{A}_k - K_k \mathcal{C}_k\right] W_k + F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) + \mathscr{K}_k - K_k H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) \tag{7.44i}$$

7.4.3 Relationship between JEKF and RPEM in the EIV framework

Recall from Section 2.5.4 that in a non-EIV setting, the major difference between the JEKF and the RPEM was the absence of the cross-coupling term $\mathscr{K}_k$ for the computation of the gradient in the former case. By including, in one way or another, an approximation of this term into the gradient computation of the JEKF, a consistent parameter estimator can be obtained (Ljung 1979). In particular, when applied to an EIV system, the JEKF can still be interpreted as a RPEM, where the term $\mathscr{K}_k$ is absent in the JEKF case. Hence, Algorithm 7.4 corresponds to Algorithm 7.1, where the latter uses a slightly modified gradient. However, the introduction of the term $\mathscr{K}_k$ into the gradient computation does not overcome the bias problem of the EIV-JEKF1 algorithm, which has been observed in Section 7.3.2. Indeed, it is shown in (Söderström 1981) (although for the case of offline PEM and by utilising a predictor based on a transfer function representation of the system) that the PEM cost function is not minimised for the true parameter values, which generally leads to biased estimates, if the PEM is directly applied to an EIV system.

Since the usage of u_{0_k} within the Jacobian design of the JEKF can yield unbiased parameter estimates as illustrated in Section 7.3.4, it would be of interest to ascertain which predictor would yield an accordingly modified gradient within the PEM framework. Or stated differently, what cost function is minimised by the parameter estimator obtained by the JEKF applied to the EIV state space system, when u_{0_k} is used to compute H_k. Let the result be captured in the following lemma.

Lemma 7.1. The parameter estimator, which is obtained from the JEKF applied to the EIV state space system, when the Jacobian H_k is computed as $H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k})$, i.e. the EIV-JEKF2 given in Algorithm 7.2, corresponds to the application of a RPEM, where the predictor is chosen as

$$\breve{y}_k(\theta) = \mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) + \mathcal{D}(\theta)u_{0_k}. \tag{7.45}$$

The corresponding prediction error is then given by

$$\begin{aligned}\breve{\varepsilon}_k(\theta) &= y_k - \breve{y}_k \\ &= y_k - \mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) - \mathcal{D}(\theta)u_{0_k}.\end{aligned} \tag{7.46}$$

The cost function minimised by the RPEM (with Λ chosen as the identity matrix) is given by

$$\breve{V}(\theta) = \frac{1}{2}\breve{\varepsilon}_k(\theta)^T\breve{\varepsilon}_k(\theta). \tag{7.47}$$

Proof. Consider $W_{k+1}(\theta)$ in (7.35), where $H(\theta, \hat{x}_{k|k-1}(\theta), u_k)$ is replaced with $H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k})$. Introducing $\breve{W}_{k+1}(\theta)$ as the quantity which uses $H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k})$ yields

$$\begin{aligned}\breve{W}_{k+1}(\theta) &= \left[\mathcal{A}(\theta) - K_k(\theta)\mathcal{C}(\theta)\right]\breve{W}_k(\theta) + F(\theta, \hat{x}_{k|k-1}(\theta), u_k) \\ &\quad + \frac{d}{d\theta}\left[K_k(\theta)\right]\varepsilon_k(\theta) - K_k(\theta)H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k}) \\ &= \frac{d}{d\theta}\left[\mathcal{A}(\theta)\right]\hat{x}_{k|k-1}(\theta) + \mathcal{A}(\theta)\breve{W}_k(\theta) + \frac{d}{d\theta}\left[\mathcal{B}(\theta)\right]u_k + \frac{d}{d\theta}\left[K_k(\theta)\right]\varepsilon_k(\theta) \\ &\quad + K_k(\theta)\left[-\frac{d}{d\theta}\left[\mathcal{C}(\theta)\right]\hat{x}_{k|k-1}(\theta) - \mathcal{C}(\theta)\breve{W}_k(\theta) - \frac{d}{d\theta}\left[\mathcal{D}(\theta)\right]u_{0_k}\right].\end{aligned} \tag{7.48}$$

Also consider the gradient (7.32) of the predictor given by (7.45), where $H(\theta, \hat{x}_{k|k-1}(\theta), u_k)$ is again replaced with $H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k})$. This gives

$$\begin{aligned}\breve{\psi}_k^T(\theta) &= \mathcal{C}(\theta)\breve{W}_k(\theta) + H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k}) \\ &= \frac{d}{d\theta}\left[\mathcal{C}(\theta)\right]\hat{x}_{k|k-1}(\theta) + \mathcal{C}(\theta)\frac{d}{d\theta}\left[\hat{x}_{k|k-1}(\theta)\right] + \frac{d}{d\theta}\left[\mathcal{D}(\theta)\right]u_{0_k}.\end{aligned} \tag{7.49}$$

A comparison of (7.48) with (7.34) as well as (7.49) with (7.32) shows that the Jacobian $H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k})$ within the JEKF is a result of choosing the prediction error as

$$\breve{\varepsilon}_k(\theta) = y_k - \mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) - \mathcal{D}(\theta)u_{0_k}. \tag{7.50}$$

This corresponds to choosing the predictor (7.45) with the gradient

$$\breve{\psi}_k^T = \mathcal{C}(\theta)\breve{W}_k(\theta) + H(\theta, \hat{x}_{k|k-1}(\theta), u_{0_k}). \tag{7.51}$$

■

Based on Lemma 7.1, it is possible to design two RPEM algorithms, analogously to the EIV-JEKF2 and EIV-JEKF3 algorithms. This development is considered in the following subsections.

7.4.4 Predictor design using u_{0_k}

As a benchmark, the RPEM with the predictor chosen as $\breve{y}_k$ given in (7.45) which corresponds to a gradient given by (7.51) is derived[3], where u_{0_k} is assumed to be available for the predictor design or, equivalently for the design of the Jacobian H_k. Of course, such an estimator is not feasible in practice, but it will serve for comparison purposes with the subsequently developed algorithms. The algorithm can be summarised as follows.

Algorithm 7.5 (EIV-RPEM2).

$$\varepsilon_k = y_k - \mathcal{C}_{k-1}\hat{x}_{k|k-1} - \mathcal{D}_{k-1}u_k \tag{7.52a}$$
$$\breve{\psi}_k = \breve{W}_k^T \mathcal{C}_{k-1}^T + H^T(\hat{\theta}_{k-1}, \hat{x}_{k|k-1}, u_{0_k}) \tag{7.52b}$$
$$R_k = R_{k-1} + \gamma_k \left[\breve{\psi}_k \breve{\psi}_k^T - R_{k-1}\right] \tag{7.52c}$$
$$\hat{\theta}_k = \hat{\theta}_{k-1} + \gamma_k R_k^{-1} \breve{\psi}_k \breve{\varepsilon}_k \tag{7.52d}$$
$$S_k = \mathcal{C}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_e^k \tag{7.52e}$$
$$K_k = \left[\mathcal{A}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_{ve}^k\right] S_k^{-1} \tag{7.52f}$$
$$\hat{x}_{k+1|k} = \mathcal{A}_k \hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k \varepsilon_k \tag{7.52g}$$
$$P_{k+1|k} = \mathcal{A}_k P_{k|k-1} \mathcal{A}_k^T + \Sigma_v^k - K_k S_k K_k^T \tag{7.52h}$$
$$\breve{W}_{k+1} = \left[\mathcal{A}_k - K_k \mathcal{C}_k\right] \breve{W}_k + F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) + \mathscr{K}_k - K_k H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_{0_k}) \tag{7.52i}$$

Note that the Kalman filter still operates with the actual innovation ε_k (rather than $\breve{\varepsilon}_k$), since only the computation of the gradient which is used for the parameter estimator is based on the predictor $\breve{y}_k$.

7.4.5 Predictor design using $\hat{u}_{0_k}$

This estimator corresponds to the EIV-JEKF3 which has been given in Algorithm 7.3. It is obtained by simply replacing u_{0_k} with its estimate $\hat{u}_{0_k}$ for the design of the Jacobian H_k. For completeness, the algorithm, denoted EIV-RPEM3, is summarised as follows.

Algorithm 7.6 (EIV-RPEM3).

$$\varepsilon_k = y_k - \mathcal{C}_{k-1}\hat{x}_{k|k-1} - \mathcal{D}_{k-1}u_k \tag{7.53a}$$
$$\hat{u}_{0_k} = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}} \mathcal{D}_k^T\right] S_{k-1}^{-1} \varepsilon_k \tag{7.53b}$$

[3]The dependency on θ is again dropped in the subsequent development for ease of notation.

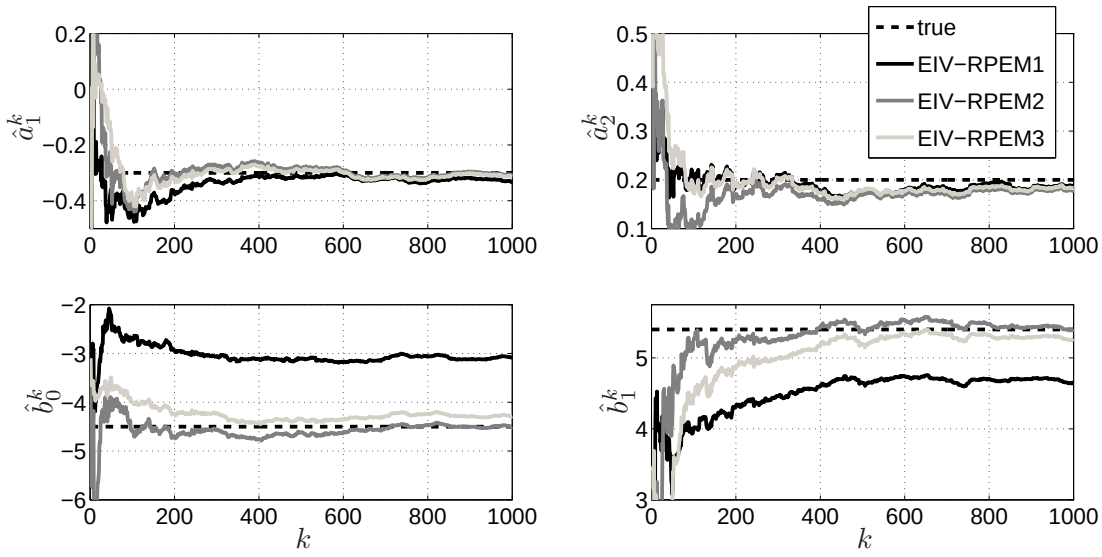


Figure 7.3: Parameter estimates obtained by the EIV-RPEM1, EIV-RPEM2 and EIV-RPEM3 algorithms.

$$\breve{\psi}_k = \breve{W}_k^T \mathcal{C}_{k-1}^T + H^T(\hat{\theta}_{k-1}, \hat{x}_{k|k-1}, \hat{u}_{0_k}) \tag{7.53c}$$

$$R_k = R_{k-1} + \gamma_k \left[\breve{\psi}_k \breve{\psi}_k^T - R_{k-1} \right] \tag{7.53d}$$

$$\hat{\theta}_k = \hat{\theta}_{k-1} + \gamma_k R_k^{-1} \breve{\psi}_k \breve{\varepsilon}_k \tag{7.53e}$$

$$S_k = \mathcal{C}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_e^k \tag{7.53f}$$

$$K_k = \left[\mathcal{A}_k P_{k|k-1} \mathcal{C}_k^T + \Sigma_{ve}^k \right] S_k^{-1} \tag{7.53g}$$

$$\hat{x}_{k+1|k} = \mathcal{A}_k \hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k \varepsilon_k \tag{7.53h}$$

$$P_{k+1|k} = \mathcal{A}_k P_{k|k-1} \mathcal{A}_k^T + \Sigma_v^k - K_k S_k K_k^T \tag{7.53i}$$

$$\breve{W}_{k+1} = \left[\mathcal{A}_k - K_k \mathcal{C}_k \right] \breve{W}_k + F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) + \mathscr{K}_k - K_k H(\hat{\theta}_k, \hat{x}_{k|k-1}, \hat{u}_{0_k}) \tag{7.53j}$$

The following subsection compares the three EIV-RPEM algorithms in a numerical example.

7.4.6 Numerical example

Consider an identical setup as in Section 7.3.2. The experiment is repeated and the three EIV-RPEM algorithms are applied to estimate the parameters of the EIV system. The resulting parameter estimates are shown in Figure 7.3. The main observation is that the estimates provided by the EIV-RPEM1, EIV-RPEM2 and EIV-RPEM3 are virtually identical to those obtained by their counterparts EIV-JEKF1, EIV-JEKF2 and EIV-JEKF3, respectively (cf. Figure 7.2). This emphasises the close relationship

between the parameter estimators obtained from the JEKF approach and the those obtained from the prediction error framework. In addition, this shows that the application of the RPEM applied to an EIV system can yield a reduced bias in the estimates, if the predictor is modified appropriately.

Remark 7.1. In the above example, the modified EIV-RPEM algorithms were able to compensate for the bias almost perfectly. The success of this method, however, depends on the quality of the noise-free input estimate. As it has been pointed out in Remark 6.2, the filter performance of the EIVKF can strongly depend on the particular system under consideration[4]. Furthermore, the feedthrough term of the system which has been considered in this example is rather dominant. It might be reasonable to assume that the bias reduction will be less efficient, if a system is chosen, where the b_0 term is less significant. In the extreme case, i.e. in the absence of a feedthrough term ($\mathcal{D} = 0$), the Jacobian H_k does not depend on the input and the modified EIV-RPEM algorithms are reduced to the standard RPEM. This means that no bias reduction is possible if $\mathcal{D} = 0$, which somewhat limits this approach.

7.5 A symmetric RPEM identification method

Although it has been shown that a modified predictor given by (7.45) can reduce the bias problem associated with the RPEM when applied for EIV system identification, it does not, however, reflect the symmetry of the EIV framework. This section introduces a further predictor, which allows the design of a cost function which minimises the distance between the measured input and output sequences and the corresponding filtered inputs and outputs, which are computed by the EIVKF.

7.5.1 Non-recursive case

The development considers first an offline or non-recursive case, where the parameter estimate is obtained by minimising a suitable cost function via a standard optimisation technique.

Define the symmetric innovation as

$$\epsilon_k(\theta) \triangleq \begin{bmatrix} y_k - \hat{y}_{0_k}(\theta) \\ u_k - \hat{u}_{0_k}(\theta) \end{bmatrix} \in \mathbb{R}^{n_y+n_u}, \tag{7.54}$$

where the estimates are computed via the EIVKF given in Algorithm 2.3 on page 42. It is then possible to design a quadratic cost function given by

$$V_\epsilon(\theta) = \frac{1}{2} E\left[\epsilon_k^T(\theta)\Lambda^{-1}\epsilon_k(\theta)\right], \tag{7.55}$$

[4]Note that Chapter 6 considers a bilinear system setup. The observations stated in Remark 6.2, however, also apply for the linear case.

which accounts for the symmetry of the EIV framework. Whilst the development of a recursive implementation is postponed until Section 7.5.3, $V_\epsilon(\theta)$ can be minimised via an appropriate optimisation routine yielding parameter estimation scheme for offline identification. In this case the parameters are obtained via

$$\hat{\theta} = \arg\min_{\theta} V_\epsilon(\theta), \tag{7.56}$$

and where the filtered inputs and outputs are computed using the EIVKF algorithm given in (7.26). For completeness, the algorithm is summarised here as follows.

Algorithm 7.7 (PEM-SYM).

$$\hat{\theta} = \arg\min_{\theta} V_\epsilon(\theta) \tag{7.57a}$$
$$\hat{x}_{k+1|k}(\theta) = \mathcal{A}(\theta)\hat{x}_{k|k-1}(\theta) + \mathcal{B}(\theta)u_k + K_k(\theta)\varepsilon_k(\theta) \tag{7.57b}$$
$$\varepsilon_k(\theta) = y_k - \mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) - \mathcal{D}(\theta)u_k \tag{7.57c}$$
$$S_k(\theta) = \mathcal{C}(\theta)P_{k|k-1}(\theta)\mathcal{C}^T(\theta) + \Sigma_e(\theta) \tag{7.57d}$$
$$K_k(\theta) = \left[\mathcal{A}(\theta)P_{k|k-1}(\theta)\mathcal{C}^T(\theta) + \Sigma_{ve}(\theta)\right] S_k^{-1}(\theta) \tag{7.57e}$$
$$P_{k+1|k}(\theta) = \mathcal{A}(\theta)P_{k|k-1}(\theta)\mathcal{A}^T(\theta) + \Sigma_v(\theta) - K_k(\theta)S_k(\theta)K_k^T(\theta) \tag{7.57f}$$
$$\hat{y}_{0_k}(\theta) = y_k - \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T\mathcal{D}^T(\theta)\right] S_k^{-1}(\theta)\varepsilon_k(\theta) \tag{7.57g}$$
$$\hat{u}_{0_k}(\theta) = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}}\mathcal{D}^T(\theta)\right] S_k^{-1}(\theta)\varepsilon_k(\theta) \tag{7.57h}$$
$$\epsilon_k(\theta) = \begin{bmatrix} y_k^T - \hat{y}_{0_k}^T(\theta) & u_k^T - \hat{u}_{0_k}^T(\theta) \end{bmatrix}^T \tag{7.57i}$$

The optimisation procedure to solve (7.57a) is generally computationally expensive, since the Kalman filter has to be applied (i.e. a Riccati equation must be solved) at each iteration, in order to obtain the symmetric innovations.

Remark 7.2. The symmetric innovation (7.57i) within the PEM-SYM algorithm is obtained via the EIVKF, which is in one-step-ahead predictor form. In the offline case, it would appear natural to assume that superior results could be obtained by making use of fixed-interval smoothing, rather than using the one-step-ahead predictor. This means that the optimal input and output estimate at time k does not only use the data up to time k (as in the prediction case), but makes use of the whole batch of available data up to time N. Using well known techniques (Anderson & Moore 1979, Section 7.4), an EIV fixed interval smoother can be derived in a straightforward manner. However, this idea is not pursued further within this thesis, but could be further work.

Note that, for comparison purposes, a similar algorithm to the PEM-SYM could be defined which minimises the squared innovations $\varepsilon_k(\theta)$ (as in the standard PEM

framework) rather than $\epsilon_k(\theta)$. For completeness, such an algorithm is summarised here as follows.

Algorithm 7.8 (PEM).

$$\hat{\theta} = \arg\min_{\theta} V(\theta) \tag{7.58a}$$

$$\hat{x}_{k+1|k}(\theta) = \mathcal{A}(\theta)\hat{x}_{k|k-1}(\theta) + \mathcal{B}(\theta)u_k + K_k(\theta)\varepsilon_k(\theta) \tag{7.58b}$$

$$\varepsilon_k(\theta) = y_k - \mathcal{C}(\theta)\hat{x}_{k|k-1}(\theta) - \mathcal{D}(\theta)u_k \tag{7.58c}$$

$$S_k(\theta) = \mathcal{C}(\theta)P_{k|k-1}(\theta)\mathcal{C}^T(\theta) + \Sigma_e(\theta) \tag{7.58d}$$

$$K_k(\theta) = \left[\mathcal{A}(\theta)P_{k|k-1}(\theta)\mathcal{C}^T(\theta) + \Sigma_{ve}(\theta)\right] S_k^{-1}(\theta) \tag{7.58e}$$

$$P_{k+1|k}(\theta) = \mathcal{A}(\theta)P_{k|k-1}(\theta)\mathcal{A}^T(\theta) + \Sigma_v(\theta) - K_k(\theta)S_k(\theta)K_k^T(\theta) \tag{7.58f}$$

Note that $V(\theta)$ is given by (7.27). Both algorithms are compared in the following example.

Example 7.1. Consider a similar setup as in Section 7.3.2, where the parameter vector is given by

$$\theta = \begin{bmatrix} -0.3 & 0.2 & -4.5 & 5.4 \end{bmatrix}^T. \tag{7.59}$$

For this example, a reduced noise environment is utilised, such that the covariance matrix of the noise sequences is given by

$$E\left[\begin{bmatrix} \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix} \begin{bmatrix} \tilde{u}_l & \tilde{y}_l & w_l \end{bmatrix}\right] = \begin{bmatrix} 0.1 & 0.5 & 0 \\ 0.5 & 2.5 & 0 \\ 0 & 0 & 0 \end{bmatrix} \delta_{kl}. \tag{7.60}$$

Algorithms 7.7 and 7.8 are applied to estimate θ using $N = 500$ samples for 100 Monte-Carlo iterations. The `fminsearch` (Nelder-Mead Simplex Method) routine of Matlab is utilised to minimise $V_\epsilon(\theta)$ and $V(\theta)$, where the initial value of $\hat{\theta}$ is chosen to be the null vector. The initial values P_0 and x_0 for the EIVKF are set as in (7.19). The mean and standard deviations of the estimates obtained are given in Table 7.1. It is observed that by making use of Algorithm 7.8, the parameters, particularly b_0 and b_1, are notably biased. This is an expected result, since in this case the PEM is directly applied to the EIV identification problem, which is known to yield biased estimates. By contrast, in the case of the symmetric innovations approach given by Algorithm 7.7, the parameters are estimated quite accurately with low standard deviations, being comparable for both approaches. Hence, it would appear that minimising the symmetric innovations could yield unbiased parameter estimates within an EIV setup. ■

	Algorithm 7.7	Algorithm 7.8
$\hat{a}_1$	$-2.96{\cdot}10^{-1} \pm 2.78{\cdot}10^{-2}$	$-3.13{\cdot}10^{-1} \pm 2.62{\cdot}10^{-2}$
$\hat{a}_2$	$2.01{\cdot}10^{-1} \pm 2.41{\cdot}10^{-2}$	$2.06{\cdot}10^{-1} \pm 2.33{\cdot}10^{-2}$
$\hat{b}_0$	$-4.51 \pm 1.33{\cdot}10^{-1}$	$-3.62 \pm 1.18{\cdot}10^{-1}$
$\hat{b}_1$	$5.38 \pm 1.46{\cdot}10^{-1}$	$4.80 \pm 1.24{\cdot}10^{-1}$

Table 7.1: Mean and standard deviation of parameter estimates for 100 Monte-Carlo iterations, comparing Algorithm 7.7 and Algorithm 7.8.

7.5.2 Analogy to Joint Output method

The PEM-SYM method given in Algorithm 7.7 strongly resembles the so called joint output method[5] (cf. Section 2.4.5), an EIV identification technique which has been proposed in (Söderström 1981). In this approach, the EIV system is reformulated as a multivariate state space system which is driven by three independent noise sources. Since the states contain the noise-free inputs and outputs, it is possible to estimate these quantities by means of a standard KF. For clarity, let such a KF applied to the multivariate state space system be denoted EIVKF-JO in the subsequent discussion. Indeed, the EIVKF-JO could be regarded as an alternative to the EIV filtering approaches that have been discussed so far within this thesis and which are based on the development described in (Diversi et al. 2005) and (Markovsky & De Moor 2005). In contrast to the EIV filters discussed in Section 2.5.2, the EIVKF-JO neither relies on the existence of a feedthrough term $\mathcal{D}$ nor on the cross-correlation between $\tilde{u}_k$ and $\tilde{y}_k$, which is believed to be one of the shortcomings of the EIVKF (cf. Remark 2.3). However, the EIVKF-JO is only able to estimate the noise-free input, since the latter is assumed to be described by an auto-regressive moving average process which is driven by white noise, i.e. u_{0_k} is assumed to be characterised by a rational spectrum (Söderström et al. 2002). Consequently, it might be beneficial to prefer either an EIVKF or an EIVKF-JO in order to filter the input and output of an EIV system, which can subsequently be utilised for the purpose of system identification. A preferred choice would certainly depend on the underlying assumption for the system and noise sequences.

Returning to the discussion comparing of the joint output method with the PEM-SYM, it is clear that the main distinction lies in the utilisation of the different EIV filters, which exploit differing assumptions to estimate u_{0_k} and y_{0_k}. A more detailed comparison of both identification techniques would be quite interesting and is identified as an avenue for potential further work.

[5] Within the literature (Söderström et al. 2002), estimators based on the joint output method are also termed maximum likelihood or prediction error approach, depending on the chosen cost function.

7.5.3 Recursive case

It is also possible to minimise $V_\epsilon(\theta)$ in a recursive manner, using the Gauss-Newton approach as outlined in Section 7.4. Indeed, only minor adjustment of the previously developed RPEM algorithms is required in order to derive a symmetric RPEM for the EIV state space system.

Recall from (7.57g)-(7.57h) that the symmetric innovation (7.54) can be re-expressed as

$$\begin{aligned}\epsilon_k(\theta) &= \begin{bmatrix}\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}^T \mathcal{D}^T(\theta) \\ \Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}} \mathcal{D}^T(\theta)\end{bmatrix} S_k^{-1}(\theta)\varepsilon_k(\theta) \\ &= T(\theta) S_k^{-1}(\theta)\varepsilon_k(\theta),\end{aligned} \tag{7.61}$$

where

$$T(\theta) \triangleq \begin{bmatrix}\Sigma_{\tilde{y}} & \Sigma_{\tilde{u}\tilde{y}}^T \\ \Sigma_{\tilde{u}\tilde{y}} & \Sigma_{\tilde{u}}\end{bmatrix} \begin{bmatrix} I \\ -\mathcal{D}^T(\theta)\end{bmatrix} \in \mathbb{R}^{(n_y+n_u)\times n_y}. \tag{7.62}$$

It is then possible to minimise the quadratic cost function $V_\epsilon(\theta)$ using a Gauss-Newton approach as outlined in Section 7.4.1. Therefore, define the gradient of the symmetric innovation as

$$-\eta_k^T(\theta) \triangleq \frac{d}{d\theta}\epsilon_k(\theta) \in \mathbb{R}^{(n_u+n_y)\times n_\theta}, \tag{7.63}$$

which, by making use of the product rule, can be computed as

$$-\eta_k^T(\theta) = J(\theta, S_k^{-1}(\theta)\varepsilon_k(\theta)) - T(\theta)S_k^{-1}(\theta)\mathscr{S}_k S_k^{-1}(\theta)\varepsilon_k(\theta) - T(\theta)S_k^{-1}(\theta)\psi_k^T(\theta), \tag{7.64}$$

with the ith column of $\mathscr{S}_k$ being given by (7.38) and where

$$J(\hat{\theta}_k, S^{-1}, \varepsilon) \triangleq \frac{d}{d\theta}\left[T(\theta)S^{-1}\varepsilon\right]\Big|_{\theta=\hat{\theta}_k} \mathbb{R}^{(n_y+n_u)\times n_\theta}, \tag{7.65}$$

is defined in a similar manner to (7.30). Consequently, the gradient of the cost function becomes

$$\frac{d}{d\theta}V_\epsilon(\theta) = -\eta_k(\theta)\Lambda^{-1}\epsilon_k(\theta). \tag{7.66}$$

Note from (7.64) that $\eta_k(\theta)$ depends on $\psi_k(\theta)$ which can be computed in a recursive fashion as outlined in Section 7.4.1. Therefore, a recursive algorithm is obtained by taking Algorithm 7.4 and modifying the computation of R_k and $\hat{\theta}_k$ in accordance with the modified gradient (7.66). By choosing $\Lambda = I$, the symmetric algorithm, which is denoted RPEM-SYM, can be summarised as follows.

Algorithm 7.9 (RPEM-SYM).

$$\varepsilon_k = y_k - \mathcal{C}_{k-1}\hat{x}_{k|k-1} - \mathcal{D}_{k-1}u_k \tag{7.67a}$$
$$\psi_k = W_k^T\mathcal{C}_{k-1}^T + H^T(\hat{\theta}_{k-1}, \hat{x}_{k|k-1}, u_k) \tag{7.67b}$$
$$\eta_k = T_{k-1}S_{k-1}^{-1}\mathscr{S}_{k-1}S_{k-1}^{-1}\varepsilon_k + T_{k-1}S_{k-1}^{-1}\psi_k^T - J(\hat{\theta}_{k-1}, S_{k-1}, \varepsilon_k) \tag{7.67c}$$
$$R_k = R_{k-1} + \gamma_k\left[\eta_k\eta_k^T - R_{k-1}\right] \tag{7.67d}$$
$$\hat{y}_{0_k}(\theta) = y_k - \left[\Sigma_{\tilde{y}} - \Sigma_{\tilde{u}\tilde{y}}{}^T\mathcal{D}_{k-1}^T(\theta)\right]S_{k-1}^{-1}\varepsilon_k \tag{7.67e}$$
$$\hat{u}_{0_k}(\theta) = u_k - \left[\Sigma_{\tilde{u}\tilde{y}} - \Sigma_{\tilde{u}}\mathcal{D}_{k-1}^T(\theta)\right]S_{k-1}^{-1}\varepsilon_k \tag{7.67f}$$
$$\epsilon_k(\theta) = \begin{bmatrix} y_k^T - \hat{y}_{0_k}^T & u_k^T - \hat{u}_{0_k}^T \end{bmatrix}^T \tag{7.67g}$$
$$\hat{\theta}_k = \hat{\theta}_{k-1} + \gamma_k R_k^{-1}\eta_k\epsilon_k \tag{7.67h}$$
$$S_k = \mathcal{C}_k P_{k|k-1}\mathcal{C}_k^T + \Sigma_e^k \tag{7.67i}$$
$$K_k = \left[\mathcal{A}_k P_{k|k-1}\mathcal{C}_k^T + \Sigma_{ve}^k\right]S_k^{-1} \tag{7.67j}$$
$$\hat{x}_{k+1|k} = \mathcal{A}_k\hat{x}_{k|k-1} + \mathcal{B}_k u_k + K_k\varepsilon_k \tag{7.67k}$$
$$P_{k+1|k} = \mathcal{A}_k P_{k|k-1}\mathcal{A}_k^T + \Sigma_v^k - K_k S_k K_k^T \tag{7.67l}$$
$$W_{k+1} = \left[\mathcal{A}_k - K_k\mathcal{C}_k\right]W_k + F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) + \mathscr{K}_k - K_k H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k) \tag{7.67m}$$

Here, the convenient notation

$$\eta_k = \eta_k(\hat{\theta}_k), \tag{7.68a}$$
$$T_k = T(\hat{\theta}_k), \tag{7.68b}$$

has been used. The performance of the algorithm is investigated in the following example.

Example 7.2. In order to compare the RPEM-SYM algorithm with the EIV-RPEM1 approach, consider a similar setup as in Example 7.1, i.e.

$$\theta = \begin{bmatrix} -0.3 & 0.2 & -4.5 & 5.4 \end{bmatrix}^T \tag{7.69}$$

and

$$E\left[\begin{bmatrix} \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix}\begin{bmatrix} \tilde{u}_l & \tilde{y}_l & w_l \end{bmatrix}\right] = \begin{bmatrix} 0.1 & 0.5 & 0 \\ 0.5 & 2.5 & 0 \\ 0 & 0 & 0 \end{bmatrix}\delta_{kl} \tag{7.70}$$

are chosen. The signal-to-noise ratio on the input and output is, respectively, given by

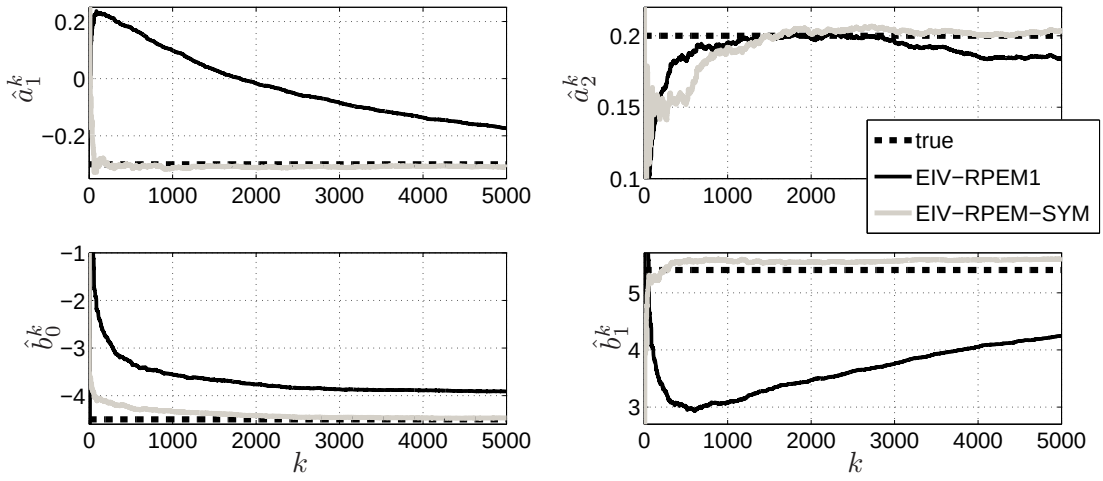


Figure 7.4: Estimates obtained from the RPEM-SYM in comparison to those obtained from the EIV-RPEM1.

29.7dB and 28.3dB. The symmetric algorithm RPEM-SYM is compared to the EIV-RPEM1 algorithm (cf. page 185) for $N = 5000$ samples. Both algorithms are initialised with

$$\hat{\theta}_0 = 0, \qquad P_0 = 100I, \qquad x_0 = 0, \qquad R_0 = 0.1I \qquad W_0 = 0. \tag{7.71}$$

In addition, a projection facility is utilised, in order to ensure that all eigenvalues of $\mathcal{A}_k - K_k\mathcal{C}_k$ lie strictly within the unit circle. The estimates obtained for both algorithms are given in Figure 7.4. It is observed that the convergence rate of the EIV-RPEM1 is rather slow and the estimates for b_0 appear to be biased. The results obtained from the RPEM-SYM, in contrast, show a rapid convergence rate and it seems that the algorithm can successfully compensate for the bias in the estimates. ■

Remark 7.3. The results obtained in Example 7.2 appear to be very promising, however, the simulation setup has been chosen with care. If lower signal-to-noise ratios are chosen, e.g. as in Section 7.3.2, it appears to be difficult to obtain satisfactory results when applying the RPEM-SYM algorithm. A potential reason for this could be that a careful initialisation of the recursive scheme is necessary and/or that a step size reduction might be required.

7.6 Concluding remarks

The extended Kalman filter for joint state and parameter estimation (JEKF) has been derived to estimate the states, the parameters as well as the noise-free inputs and outputs of an errors-in-variables (EIV) state space system. A simulation has revealed, however, that the parameter estimates obtained are biased. Since the parameter estimator, which is obtained by applying the JEKF, can be interpreted as a recursive

Alg.	Name	Description
7.1	EIV-JEKF1	Direct application of the JEKF for EIV state space systems. Produces biased parameter estimates in presence of input noise.
7.2	EIV-JEKF2	Modified JEKF algorithm. Utilises u_{0_k} for design of Jacobian H_k. Can reduce the bias in parameter estimates.
7.3	EIV-JEKF3	Modified JEKF algorithm. Utilises $\hat{u}_{0_k}$ for design of Jacobian H_k. Can reduce the bias in parameter estimates.
7.4	EIV-RPEM1	'True' RPEM applied to an EIV state space system. Produces biased parameter estimates in presence of input noise.
7.5	EIV-RPEM2	Modified predictor. Analogue to EIV-JEKF2. Utilises u_{0_k} for design of Jacobian H_k. Can reduce the bias in parameter estimates.
7.6	EIV-RPEM3	Modified predictor. Analogue to EIV-JEKF3. Utilises $\hat{u}_{0_k}$ for design of Jacobian H_k. Can reduce the bias in parameter estimates.
7.7	PEM-SYM	Offline PEM technique, minimising quadratic cost function of symmetrically defined innovations.
7.8	PEM	'True' (offline) PEM applied to an EIV state space system. Produces biased parameter estimates in presence of input noise.
7.9	RPEM-SYM	Recursive implementation of PEM-SYM algorithm.

Table 7.2: Overview of developed algorithms for Chapter 7 (Alg. stands for Algorithm).

prediction error method (RPEM), this is not a surprising result, since it is known within the literature, that a direct application of the prediction error method (PEM) to the EIV case does not yield consistent estimates. Making use of the filtered input for the design of the Jacobians which are used within the JEKF design yields a modified version of the JEKF. Using this algorithm, it is possible to, at least partly, compensate for the resulting bias in the estimates. By deriving the 'true' RPEM for the EIV state space system and using these results to analyse the modified JEKF procedure, it is revealed that the usage of the filtered input in the Jacobian design corresponds to a modified predictor, which is able to reduce the bias of the EIV system parameters in the presence of a feedthrough term. Whilst this modified predictor is able to reduce the bias in the estimates, it does not account for the symmetric structure of the EIV framework. Consequently, a novel algorithm is developed, which uses the filtered inputs and outputs to define symmetric innovations. The corresponding predictor then allows the design of a symmetric (non-recursive) PEM, which resembles the joint output method, a well known EIV identification technique. Moreover, an algorithm for recursively solving the identification problem, which is based on the RPEM design, is also developed. An overview of the algorithms, which have been developed within this Chapter, is provided in Table 7.2.

The thread which has been followed throughout this chapter, i.e. the idea of combining filtering and estimation, is certainly not novel, but a common approach within the system identification literature (cf. e.g. Ljung 1999, Young 1984, Young et al. 2001). Filtering the data explicitly or implicitly (as realised in the joint output method) can be

utilised in order to affect the asymptotic bias in the parameter estimates. The novelty within this chapter is that usage of the EIVKF has been made in order to filter the inputs and outputs. Consequently, the limitations regarding the EIVKF, as mentioned in Remark 2.3, also apply for the algorithms of this chapter, which has been discussed in Remark 7.1. Therefore, it would be of interest to investigate, under which exact requirements the EIVKF can provide reasonable estimates. With 'reasonable', it is meant, that the EIVKF is, on average, able to effectively remove noise from the measured inputs and outputs, as indicated by a positive value of the performance criteria (7.20). In addition, an EIV fixed interval smoother could yield superior results with respect to the one-step-ahead predictor, which is utilised by the EIVKF considered in this thesis. These aspects are deemed to be interesting avenues for potential further work. In addition, the convergence and consistency properties of the developed algorithms have not yet been investigated. For potential practical applications, however, such an analysis is of major interest and can also identified as potential future work. This also applies to the relationship between the new PEM-SYM algorithm and the joint output method, which has been discussed, but requires a more thorough analysis which is also identified as potential further work.

Chapter 8

Conclusions & further work

8.1 Conclusions

Prompted by a desire to exploit the errors-in-variables system identification and filtering approaches to encompass online realisation of the techniques, this thesis has focused attention on the development of pragmatically applicable algorithms for potential online implementation. As outlined in Chapter 1, there are two main threads, namely:

1. Errors-in-variables identification, where the emphasis has been focused on the development of recursive algorithms to realise the Frisch scheme, as well as exploiting the similarities to the extended bias compensating least squares framework.

2. Errors-in-variables filtering where the emphasis is aimed at developing recursive algorithms for reconstructing the noise-free input and noise-free output signals, which may be subsequently utilised for errors-in-variables identification.

The development of the algorithms within this thesis is considered to be timely in that there exists significant immediate potential for deployment in a range of industrial/commercial settings where models with inherently unbiased parameters can be utilised to advantage.

This Chapter summarises the main outcomes of the research documented in this thesis. The two threads of research are separately discussed in Sections 8.1.1 and 8.1.2, with reference to the chapters where details can be found. The main contributions are summarised in descending order of significance in Section 8.1.3. Finally, avenues for further work, as identified throughout the thesis, are summarised and listed in Section 8.2 in order of importance in the context of this work.

8.1.1 Recursive Frisch scheme identification

Algorithms to recursively approximate the Frisch scheme estimates of linear time-invariant single-input single-output dynamic errors-in-variables systems have been developed. The estimates of the model parameters and the input/output measurement noise

variances are updated via gradient-based techniques, which are developed based on the offline Frisch scheme equations for the white noise case. Two novel recursive algorithms have been proposed in Chapter 3, which allow the update of a Frisch scheme model as new data becomes available. This extends the applicability of the Frisch scheme towards cases, which often rely on an online implementation, such as fault detection, adaptive signal processing and adaptive errors-in-variables filtering. It has been shown that the recursive algorithms can successfully approximate the offline Frisch scheme solution and even retain the Frisch-character, the unique feature of the Frisch scheme. A detailed analysis of the computational complexity of the developed algorithms has been carried out. This has prompted, in Chapter 4, the development of two additional fast recursive Frisch scheme algorithms, which reduce the computational complexity from cubic to second order. The development of the fast algorithms further increases the applicability range of the recursive Frisch scheme algorithms towards cases where only limited computational resources are available and the use of complex algorithm implementations is prohibitive.

In addition to the development of recursive approaches based on the offline Frisch scheme, recursive algorithms for the extended bias compensating least squares identification problem have also been developed in Chapter 5 for the white noise case. Since the Frisch scheme using the Yule-Walker model selection criterion can be interpreted within this framework, the developed algorithms provide an alternative to the recursive methods based on the offline Frisch scheme, hence enriching the portfolio of available algorithms. A bilinear parametrisation concept has been exploited to derive a type of recursive algorithm based on a two-step estimation approach leading to a computationally attractive implementation. Moreover, a recursive implementation of the variable projection method, which is also known as the nonlinear separable least squares technique, has been considered. This not only yields numerically sound algorithms for the identification problem, but also allows the generalisability towards more complex bias compensating settings, such as the coloured output noise case. Thus, the algorithms developed within this framework, find wider applicability beyond the scope of the identification via the Frisch scheme approach. In addition, an extensive simulation study has provided a detailed comparison between the recursive algorithms developed within the extended bias compensation framework and those based on the offline Frisch scheme.

As well as the case of white output noise, a modified scenario whereby the output noise is correlated in time (that is coloured output noise), a setup which is likely to be more realistic in many practical applications, has been considered. Two novel recursive algorithms have been proposed in Chapter 4, which have been developed from existing offline identification schemes. Whilst the first algorithm follows a basic approach involving the application of Newton's method, the second algorithm leads to a more straightforward and less computationally demanding scheme, which exploits the

special structure of the underlying problem.

8.1.2 Errors-in-variables filtering

The theory of errors-in-variables Kalman filtering, which has only been considered for linear systems within the literature, has been extended in Chapter 6 to encompass a class of bilinear systems, an appealing and commonly utilised subset of nonlinear systems. The significance of this contribution is twofold: Firstly, it can be considered as a piece of pioneering work of extending the theory of errors-in-variables filtering towards more practically oriented, yet correspondingly more sophisticated nonlinear system structures. Since bilinear systems are closely related to linear systems, it appears to be a most natural step to begin such an extension with this relatively simple class of nonlinear systems. Secondly, since bilinear models are widely used within numerous versatile areas, this contribution also lays the foundation for a practical application of errors-in-variables filtering to the wider realm of real-world problems. Whilst the optimal algorithm is shown to be infeasible in the bilinear errors-in-variables case, the outcome of this work is that of a collection of four suboptimal algorithms from which a user can choose the most suitable filter structure depending on the underlying application. Due to the close relationship of bilinear and linear systems, the developed filters are all based on the well known Kalman filter theory, hence they are relatively easy to understand and to apply.

In addition to the bilinear errors-in-variables problem, Chapter 7 considers the idea of combining (linear) errors-in-variables filtering and system identification, i.e. to utilise the filtered inputs and outputs within the identification algorithm, in order to reduce and eventually overcome the asymptotic bias of otherwise inconsistent estimators. Whilst filtering for identification is commonly used within the literature to affect the bias distribution of the estimates, the usage of the errors-in-variables Kalman filter for this purpose has, to the author's knowledge, not yet been considered. The basis for the system identification procedure has been the recursive prediction error method (either in approximate form given by the extended Kalman filter for joint state and parameter estimation, or in its 'true'/direct form), which is known to yield biased estimates in the presence of input measurement noise. A modified predictor has been derived, which is shown (via simulation) to radically reduce the bias in the parameter estimates of an errors-in-variables state space system. The chapter culminates, by proposing a modified symmetric predictor, which can be implemented in an offline or online manner. In addition, a Monte-Carlo simulation has indicated that the technique may yield unbiased estimates. Finally, the similarities to the joint output method have been discussed. The outcome is thus a novel errors-in-variables identification technique, which provides a basis for discussion and eventual future research.

8.1.3 Contributions in descending order of significance

A summary of the developments within this thesis are given as follows in a descending order with respect to their considered significance.

1. Recursive algorithms based on the offline Frisch scheme using the Yule-Walker model selection criterion have been developed (Chapter 3).
2. A symmetric prediction error method for errors-in-variables identification which utilises errors-in-variables filtering techniques has been proposed (Chapter 7).
3. Based on the proposed recursive Frisch scheme algorithms, computationally less expensive procedures have been developed (Chapter 4).
4. Algorithms for bilinear errors-in-variables filtering have been developed (Chapter 6).
5. Recursive algorithms for the extended bias compensating least squares approach have been proposed (Chapter 5).
6. Recursive Frisch scheme algorithms for the coloured output noise case have been proposed (Chapter 4).

8.2 Further work

The developments within this thesis have identified several topics for potential further work.

> In the time available, it did not prove possible to include any real world examples, although several were considered. There is an urgent requirement, therefore, to evaluate all of the proposed algorithms on some real application data. Work in this regard is proceeding and will be reported in future publications.

As regards further algorithmic developments, additional research on the recursive Frisch scheme should include:

- The development of alternative algorithms to replace the recursive bias compensating least squares technique for the computation of the parameter vector. In particular, the use of matrix factorisations could be an interesting direction.
- The use of other subspace tracking algorithms rather than the conjugate gradient method might provide attractive alternatives.
- An extension of the developed recursive Frisch scheme algorithms when use is made of the covariance match criterion is possible and is an interesting aspect for further work.

- A thorough mathematical analysis of the convergence and consistency aspects of (some of) the developed algorithms would be desirable and could provide another interesting topic of further work.

- All of the recursive algorithms should be considered from the point of view of time variable parameter estimation since this is one of the major reasons for exploiting recursive algorithms. In addition to simple exponential data weighting of the kind considered in Appendix B, more sophisticated and flexible optimal approaches need to be evaluated, as discussed for example in (Young 1999, Young 2000, Young 2002). These include stochastic modelling of the parameter variations and optimisation of the associated hyper-parameters, as well as the implementation of recursive fixed interval smoothing to allow for improved offline estimation and the removal of the estimation lag that affects online 'filtering' recursions.

For the errors-in-variables filtering, the following aspects are considered to provide topics for potential further work:

- The development of robust errors-in-variables filters for bilinear systems could be a topic of interest.

- The extension of errors-in-variables filtering techniques towards more general non-linear systems, other than bilinear, might be desirable. Within this context, Bayesian approaches for bilinear errors-in-variables filtering might lead to improved performance for the non-Gaussian case.

- The theory of (linear) errors-in-variables filtering techniques could be extended to deal with stochastic parameters or coloured noise setups which would be another direction for further work.

- An investigation of the relationship between errors-in-variables Kalman filtering and classical Kalman filters, which can estimate the noise-free input when the latter is described by a rational spectrum, would be illustrative. In particular, it would be interesting to develop the optimal filter for the case when the input has a rational spectrum and, in addition, when a feedthrough term and/or cross-correlation between the input and output measurement noise is present (which forms the basis for the errors-in-variables Kalman filter).

- In connection with the previous point, a detailed comparison of the joint output method and symmetric prediction error method would be of interest. In addition, a convergence and consistency analysis for the latter would also be desirable.

- Rather than the Kalman filter in one-step ahead prediction form, a fixed interval smoother could be used for the symmetric prediction error method in the offline case. This is likely to give some improvement with respect to estimation performance and is, therefore, of potential interest.

Appendices

Appendix A

RBCLS derivation

Equation (2.1) can be rewritten as

$$A(q^{-1})y_i - B(q^{-1})u_i = A(q^{-1})\tilde{y}_i - B(q^{-1})\tilde{u}_i \triangleq e_i, \tag{A.1}$$

where the residual e_i is the difference of two moving average processes. This allows the formulation of a linear regression problem

$$y_i = \varphi_i^T \theta + e_i \tag{A.2}$$

and the application of the least squares (LS) estimator. It is well known that the LS estimate

$$\hat{\theta}_k^{\mathrm{LS}} = \left[\sum_{i=1}^{k} \varphi_i \varphi_i^T\right]^{-1} \sum_{i=1}^{k} \varphi_i y_i \tag{A.3}$$

is asymptotically biased in the presence of measurement noise. An explicit expression for the bias can be obtained by substituting (A.2) in (A.3) which yields

$$\hat{\theta}_k^{\mathrm{LS}} = \theta + \left[\sum_{i=1}^{k} \varphi_i \varphi_i^T\right]^{-1} \sum_{i=1}^{k} \varphi_i e_i. \tag{A.4}$$

Making use of the fact that $e_i = -\tilde{\varphi}_i^T \theta + \tilde{y}_i$, and dividing by k it follows that

$$\frac{1}{k}\sum_{i=1}^{k} \varphi_i \varphi_i^T \left(\hat{\theta}_k^{\mathrm{LS}} - \theta\right) = -\frac{1}{k}\sum_{i=1}^{k} \varphi_i \tilde{\varphi}_i \theta + \frac{1}{k}\sum_{i=1}^{k} \varphi_i \tilde{y}_i, \tag{A.5}$$

which becomes, in the asymptotic case, i.e. for $k \to \infty$

$$\Sigma_\varphi \left(\theta^{\mathrm{LS}} - \theta\right) = -\begin{bmatrix} \sigma_{\tilde{y}} I_{n_a} & 0 \\ 0 & \sigma_{\tilde{u}} I_{n_b} \end{bmatrix} \theta \tag{A.6}$$

or, equivalently,

$$\theta = \theta^{\mathrm{LS}} + \Sigma_{\varphi}^{-1} \begin{bmatrix} \sigma_{\tilde{y}} I_{n_a} & 0 \\ 0 & \sigma_{\tilde{u}} I_{n_b} \end{bmatrix} \theta, \tag{A.7}$$

where Σ_{φ} is obtained by deleting the first row and column of $\Sigma_{\bar{\varphi}}$.

Equation (A.7) gives rise to a recursive form and, if the noise variances are known (or estimated), it is possible to apply the RLS estimator and compensate for the bias at each time step k. This gives the update equation

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{LS}} + \left[\hat{\Sigma}_{\varphi}^{k}\right]^{-1} \Sigma_{\tilde{\varphi}}(\hat{\sigma}_k)\hat{\theta}_{k-1} \tag{A.8}$$

with the noise compensation matrix

$$\Sigma_{\tilde{\varphi}}(\hat{\sigma}_k) = \begin{bmatrix} \hat{\sigma}_{\tilde{y}}^{k} I_{n_a} & 0 \\ 0 & \hat{\sigma}_{\tilde{u}}^{k} I_{n_b} \end{bmatrix}. \tag{A.9}$$

Appendix B

Recursive update of covariance matrices

This appendix reviews the recursive update equation for covariance matrix estimates for both, equally and exponentially weighted data.

B.1 Equally weighted data

A commonly used estimator for a covariance matrix of two general vectors a_i and b_i is defined by

$$\hat{\Sigma}_{ab}^{k} \triangleq \frac{1}{k}\sum_{i=1}^{k} a_i b_i^T. \tag{B.1}$$

A recursive version of this estimator is given by

$$\hat{\Sigma}_{ab}^{k} = \frac{1}{k}\left[\sum_{i=1}^{k-1} a_i b_i^T + a_k b_k^T\right], \tag{B.2}$$

$$= \frac{1}{k}\left[(k-1)\hat{\Sigma}_{ab}^{k-1} + a_k b_k^T\right], \tag{B.3}$$

$$= \hat{\Sigma}_{ab}^{k-1} + \frac{1}{k}\left[a_k b_k^T - \hat{\Sigma}_{ab}^{k-1}\right]. \tag{B.4}$$

B.2 Exponentially weighted data

In order to allow an algorithm to be adaptive, an exponential form of data weighting may be utilised (Ljung 1999, Sec. 11.2). This can be realised by giving the ith measurement at time k the weighting β_i^k, which satisfies the properties

$$\beta_i^k = \lambda_k \beta_i^{k-1} \quad \text{for } 0 \leq i < k-1, \tag{B.5}$$

$$\beta_k^k = 1, \tag{B.6}$$

where $0 < \lambda_k \leq 1$ is a potentially time-varying forgetting factor. The properties of the weighting leads to the recursive relation

$$\sum_{i=1}^{k} \beta_k^i a_i b_i^T = \sum_{i=1}^{k-1} \beta_k^i a_i b_i^T + a_k b_k^T, \tag{B.7}$$

$$= \lambda_k \sum_{i=1}^{k-1} \beta_{k-1}^i a_i b_i^T + a_k b_k^T. \tag{B.8}$$

Due to the individual weighting of each summand in (B.7), the accordingly weighted covariance matrix estimate cannot simply use the scaling $1/k$ as is (B.1). Instead, a weighted arithmetic mean is to be computed which is given by

$$\bar{\Sigma}_{ab}^k = \gamma_k \sum_{i=1}^{k} \beta_k^i a_i b_i^T, \tag{B.9}$$

where γ_k denotes a normalising gain given by

$$\gamma_k = \frac{1}{\sum_{i=1}^{k} \beta_i^k}. \tag{B.10}$$

Note that in the case of no adaptivity, i.e. $\lambda_k = 1$ for all k, the data weighting becomes $\beta_i^k = 1$ for all $0 \leq i < k$ which results in $\gamma_k = 1/k$. In this case (B.9) becomes (B.1), i.e. the standard estimator for covariance matrices. Using (B.8) and (B.9), the recursive update of the $\bar{\Sigma}_{ab}^k$ is given by

$$\bar{\Sigma}_{ab}^k = \gamma_k \sum_{i=1}^{k} \beta_k^i a_i b_i^T, \tag{B.11}$$

$$= \gamma_k \left[\lambda_k \sum_{i=1}^{k-1} \beta_{k-1}^i a_i b_i^T + a_k b_k^T \right], \tag{B.12}$$

$$= \gamma_k \left[\lambda_k \frac{1}{\gamma_{k-1}} \bar{\Sigma}_{ab}^{k-1} + a_k b_k^T \right]. \tag{B.13}$$

Exploiting the fact that the normalising gain satisfies the recursive relationship

$$\frac{\lambda_k}{\gamma_{k-1}} = \frac{1}{\gamma_k} - 1, \tag{B.14}$$

equation (B.13) simplifies to

$$\bar{\Sigma}_{ab}^k = \bar{\Sigma}_{ab}^{k-1} + \gamma_k \left[a_k b_k^T - \bar{\Sigma}_{ab}^{k-1} \right]. \tag{B.15}$$

Appendix C

Linearisation of the Frisch scheme equations

This appendix reviews the linearisation of the Frisch scheme equations as outlined in (Söderström 2007*a*). Dropping the time index for the ease of notation, recall that the basic Frisch equations are given by

$$\left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\hat{\sigma})\right)\hat{\theta} = \hat{\xi}_{\varphi y}, \tag{C.1a}$$

$$\hat{\sigma}_{\tilde{y}} = \lambda_{\min}\left(\hat{\Sigma}_{\bar{\varphi}_y} - \hat{\Sigma}_{\bar{\varphi}_y\varphi_u}\left[\hat{\Sigma}_{\varphi_u} - \hat{\sigma}_{\tilde{u}} I_{n_b}\right]^{-1}\hat{\Sigma}_{\varphi_u\bar{\varphi}_y}\right). \tag{C.1b}$$

The linearisation is carried out around the value

$$\vartheta^* \triangleq \begin{bmatrix}\theta^* \\ \sigma^*_{\tilde{y}} \\ \sigma^*_{\tilde{u}}\end{bmatrix}. \tag{C.2}$$

C.1 Linearisation of θ-equation

From (C.1a), one obtains

$$\begin{aligned}\hat{\theta} - \theta^* &= \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\hat{\sigma})\right)^{-1}\hat{\xi}_{\varphi y} - \theta^* \\ &= \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\hat{\sigma})\right)^{-1}\left(\hat{\xi}_{\varphi y} - \left(\hat{\Sigma}_\varphi - \begin{bmatrix}\hat{\sigma}_{\tilde{y}} I_{n_a} & 0 \\ 0 & \hat{\sigma}_{\tilde{u}} I_{n_b}\end{bmatrix}\right)\theta^*\right) \\ &\approx \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)^{-1}\left(\hat{\xi}_{\varphi y} - \hat{\Sigma}_\varphi\theta^* + \begin{bmatrix}\hat{\sigma}_{\tilde{y}} a^* \\ \hat{\sigma}_{\tilde{u}} b^*\end{bmatrix}\right),\end{aligned} \tag{C.3}$$

where it is assumed that $\hat{\vartheta}$ is close to the point of linearisation ϑ^*, in order to allow for the approximation

$$\left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\hat{\sigma})\right)^{-1} \approx \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)^{-1}. \tag{C.4}$$

Further modification of (C.3) gives

$$\begin{aligned}
\hat{\theta} - \theta^* &= \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)^{-1} \left(\frac{1}{k}\sum_{i=1}^{k} \varphi_i y_i - \frac{1}{k}\sum_{i=1}^{k} \varphi_i \varphi_i^T \theta^* + \begin{bmatrix} \hat{\sigma}_{\tilde{y}} a^* \\ \hat{\sigma}_{\tilde{u}} b^* \end{bmatrix}\right) \\
&= \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)^{-1} \left(\frac{1}{k}\sum_{i=1}^{k} \varphi_i \left(y_i - \varphi_i^T \theta^*\right) + \begin{bmatrix} \hat{\sigma}_{\tilde{y}} a^* \\ \hat{\sigma}_{\tilde{u}} b^* \end{bmatrix} + \begin{bmatrix} \sigma^*_{\tilde{y}} a^* \\ \sigma^*_{\tilde{u}} b^* \end{bmatrix} - \begin{bmatrix} \sigma^*_{\tilde{y}} a^* \\ \sigma^*_{\tilde{u}} b^* \end{bmatrix}\right) \\
&= \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)^{-1} \left(\frac{1}{k}\sum_{i=1}^{k} \varphi_i \varepsilon(\theta^*) + \begin{bmatrix} (\hat{\sigma}_{\tilde{y}} - \sigma^*_{\tilde{y}}) a^* \\ (\hat{\sigma}_{\tilde{u}} - \sigma^*_{\tilde{u}}) b^* \end{bmatrix} + \begin{bmatrix} \sigma^*_{\tilde{y}} a^* \\ \sigma^*_{\tilde{u}} b^* \end{bmatrix}\right),
\end{aligned} \tag{C.5}$$

which yields the linearisation result given in (Söderström 2007*a*) given by

$$\begin{aligned}
\left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)\left(\hat{\theta} - \theta^*\right) - \begin{bmatrix} a^* \\ 0 \end{bmatrix}(\hat{\sigma}_{\tilde{y}} - \sigma^*_{\tilde{y}}) - \begin{bmatrix} 0 \\ b^* \end{bmatrix}(\hat{\sigma}_{\tilde{u}} - \sigma^*_{\tilde{u}}) = \\
\frac{1}{k}\sum_{i=1}^{k} \varphi_i \varepsilon(\theta^*) + \begin{bmatrix} \sigma^*_{\tilde{y}} a^* \\ \sigma^*_{\tilde{u}} b^* \end{bmatrix}.
\end{aligned} \tag{C.6}$$

C.1.1 Sensitivity derivatives

From (C.3) it is straightforward to compute

$$\frac{\partial \hat{\theta}}{\partial \hat{\sigma}_{\tilde{u}}} \approx \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)^{-1} \begin{bmatrix} 0 \\ b^* \end{bmatrix}, \tag{C.7a}$$

$$\frac{\partial \hat{\theta}}{\partial \hat{\sigma}_{\tilde{y}}} \approx \left(\hat{\Sigma}_\varphi - \Sigma_{\tilde{\varphi}}(\sigma^*)\right)^{-1} \begin{bmatrix} a^* \\ 0 \end{bmatrix}. \tag{C.7b}$$

C.2 Linearisation of $\lambda_{\min}$-equation

The idea is to linearise the $\lambda_{\min}$-equation around ϑ^* using perturbation theory.

Assume that ϑ^* satisfies the compensated normal equations

$$\begin{bmatrix} \hat{\Sigma}_{\bar{\varphi}_y} - \Sigma_{\tilde{\varphi}_y}(\sigma^*_{\tilde{y}}) & \hat{\Sigma}_{\bar{\varphi}_y \varphi_u} \\ \hat{\Sigma}_{\varphi_u \bar{\varphi}_y} & \hat{\Sigma}_{\varphi_u} - \Sigma_{\tilde{\varphi}_y}(\sigma^*_{\tilde{u}}) \end{bmatrix} \begin{bmatrix} \bar{a}^* \\ b^* \end{bmatrix} = 0 \tag{C.8}$$

which are rewritten for ease of notation as

$$\begin{bmatrix} A - \sigma^*_{\tilde{y}} I_{n_a} & C \\ C^T & D_* \end{bmatrix} \begin{bmatrix} \bar{a}_* \\ b_* \end{bmatrix} = 0. \tag{C.9}$$

Similarly, introduce

$$\begin{bmatrix} A - \hat{\sigma}_{\tilde{y}} I_{n_a} & C \\ C^T & \hat{D} \end{bmatrix} \begin{bmatrix} \hat{\bar{a}} \\ \hat{b} \end{bmatrix} = 0. \tag{C.10}$$

Note that $\sigma^*_{\tilde{y}}$ and $\hat{\sigma}_{\tilde{y}}$ are obtained via

$$\sigma^*_{\tilde{y}} = \lambda_{\min}\left(B_*\right), \tag{C.11a}$$

$$\hat{\sigma}_{\tilde{y}} = \lambda_{\min}\left(\hat{B}\right), \tag{C.11b}$$

where

$$B_* = A - C D_*^{-1} C^T, \tag{C.12a}$$

$$\hat{B} = A - C \hat{D}^{-1} C^T. \tag{C.12b}$$

Now, the result of perturbation theory states that

$$\lambda_{\min}(\hat{B}) = \lambda_{\min}(B_* + \Delta B) \approx \lambda_{\min}(B_*) + \frac{\bar{a}_*^T \Delta B \bar{a}_*}{\bar{a}_*^T \bar{a}_*}, \tag{C.13}$$

where

$$\Delta B = \hat{B} - B_* = C\left(D_*^{-1} - \hat{D}^{-1}\right) C^T. \tag{C.14}$$

Consequently, one can write

$$\begin{aligned} \hat{\sigma}_{\tilde{y}} - \sigma^*_{\tilde{y}} &\approx \lambda_{\min}(B_*) + \frac{\bar{a}_*^T \Delta B \bar{a}_*}{\bar{a}_*^T \bar{a}_*} - \lambda_{\min}(B_*) \\ &= \frac{\bar{a}_*^T}{\bar{a}_*^T \bar{a}_*} \left(\Delta B\right) \bar{a}_*. \end{aligned} \tag{C.15}$$

Expanding ΔB as

$$\begin{aligned} \Delta B &= C D_*^{-1} C^T - C \hat{D}^{-1} C^T \\ &= C D_*^{-1} \left(\hat{D} - D_*\right) \hat{D}^{-1} C^T \end{aligned} \tag{C.16}$$

and introducing

$$\hat{D} = \bar{D} - \hat{\sigma}_{\tilde{u}} I_{n_b}, \tag{C.17a}$$

$$D_* = \bar{D} - \sigma^*_{\tilde{u}} I_{n_b}, \tag{C.17b}$$

gives

$$\hat{D} - D_* = \bar{D} - \hat{\sigma}_{\tilde{u}} I_{n_b} - \bar{D} + \sigma^*_{\tilde{u}} I_{n_b} = \sigma^*_{\tilde{u}} I_{n_b} - \hat{\sigma}_{\tilde{u}} I_{n_b}, \tag{C.18}$$

and Equation (C.15) becomes

$$\bar{a}_*^T \bar{a}_* \left(\hat{\sigma}_{\tilde{y}} - \sigma_{\tilde{y}}^*\right) \approx -\bar{a}_*^T C D_*^{-1} \left(\hat{\sigma}_{\tilde{u}} - \sigma_{\tilde{u}}^*\right) \hat{D}^{-1} C^T \bar{a}_*. \tag{C.19}$$

From the lower part of (C.9), one obtains

$$b_* = -D_*^{-1} C^T a_*, \tag{C.20}$$

and by assuming that $b_* \approx \hat{D}^{-1} C^T a_*$, (C.19) becomes

$$\bar{a}_*^T \bar{a}_* \left(\hat{\sigma}_{\tilde{y}} - \sigma_{\tilde{y}}^*\right) \approx -b_*^T b_* \left(\hat{\sigma}_{\tilde{u}} - \sigma_{\tilde{u}}^*\right). \tag{C.21}$$

C.2.1 Sensitivity derivatives

From (C.21) it is straightforward to compute

$$\frac{d\hat{\sigma}_{\tilde{y}}}{d\hat{\sigma}_{\tilde{u}}} \approx -\frac{b_*^T b_*}{\bar{a}_*^T \bar{a}_*}. \tag{C.22}$$

Appendix D

Derivatives for RFSCON1

D.1 First order derivative of V_1^k

Denoting $(\cdot)'$ the derivative w.r.t. $\hat{\sigma}_{\tilde{u}}^k$ and introducing

$$f_k \triangleq G_k^T \hat{\xi}_{\delta y}^k, \qquad F_k \triangleq G_k^T G_k, \tag{D.1}$$

it holds that

$$f_k' = -\begin{bmatrix} 0 \\ \hat{\xi}_{\varphi_u y}^k \end{bmatrix}, \tag{D.2a}$$

$$F_k' = \begin{bmatrix} 0 & \hat{\Sigma}_{\varphi_u \varphi_y}^{k\ T} \\ \hat{\Sigma}_{\varphi_u \varphi_y}^k & 2\hat{\sigma}_{\tilde{u}}^{k-1} I_{n_b} - 2\hat{\Sigma}_{\varphi_u}^k \end{bmatrix}, \tag{D.2b}$$

$$F_k^{-1\prime} = -F_k^{-1} F_k' F_k^{-1} \tag{D.2c}$$

and the first order derivative is given by

$$\begin{aligned} V_1^{k\prime} &= -\left(f_k^T F_k^{-1} f_k\right)' \\ &= -f_k'^{\,T} F_k^{-1} f_k - f_k^T F_k^{-1\prime} f_k - f_k^T F_k^{-1} f_k'. \end{aligned} \tag{D.3}$$

D.2 Second order derivative of V_1^k

Utilising the product rule, the second order derivative is given by

$$\begin{aligned} V_1^{k\prime\prime} &= -f_k'^{\,T} F_k^{-1\prime} f_k - f_k'^{\,T} F_k^{-1} f_k' \\ &\quad - f_k'^{\,T} F_k^{-1\prime} f_k - f_k^T F_k^{-1\prime\prime} f_k - f_k^T F_k^{-1\prime} f_k' \\ &\quad - f_k'^{\,T} F_k^{-1} f_k' - f_k^T F_k^{-1\prime} f_k' \\ &= -2 f_k'^{\,T} F_k^{-1\prime} f_k - 2 f_k'^{\,T} F_k^{-1} f_k' \\ &\quad - f_k^T F_k^{-1\prime\prime} f_k - 2 f_k^T F_k^{-1\prime} f_k' \end{aligned} \tag{D.4}$$

with

$$F_k^{-1\,\prime\prime} = -\,F_k^{-1\,\prime}\,F_k'F_k^{-1} - F_k^{-1}F_k''F_k^{-1} - F_k^{-1}F_k'\,F_k^{-1\,\prime}, \tag{D.5a}$$

$$F_k'' = \begin{bmatrix} 0 & 0 \\ 0 & 2I_{n_b} \end{bmatrix}. \tag{D.5b}$$

D.3 Derivative of $\hat{\varsigma}_k$

The idea is to linearise the Frisch equation (2.82) using perturbation theory (as in Appendix C.2), in order to approximate the derivative of $\hat{\varsigma}_k$ w.r.t. α_k. The derivation here is conceptually similar to that given in Appendix II.B of (Söderström 2007*a*), but with the linearisation carried out around $\hat{\vartheta}_{k-1}$ rather than the true parameters ϑ_0.

Assume that at time instance $k-1$, $\hat{\vartheta}_{k-1}$ satisfies the extended compensated normal equations

$$\begin{bmatrix} \hat{\Sigma}_{\bar{\varphi}_y}^{k-1} - \hat{\Sigma}_{\bar{\varphi}_y}^{k-1} & \hat{\Sigma}_{\bar{\varphi}_y\varphi_u}^{k-1} \\ \hat{\Sigma}_{\varphi_u\bar{\varphi}_y}^{k-1} & \hat{\Sigma}_{\varphi_u}^{k-1} - \hat{\sigma}_{\tilde{u}}^{k-1} I_{n_b} \end{bmatrix} \begin{bmatrix} \hat{\bar{a}}_{k-1} \\ \hat{b}_{k-1} \end{bmatrix} = 0 \tag{D.6}$$

which are rewritten for ease of notation as

$$\begin{bmatrix} \mathfrak{A} - \mathfrak{B} & \mathfrak{C} \\ \mathfrak{C}^T & \mathfrak{D} - \hat{\sigma}_{\tilde{u}}^{k-1} I \end{bmatrix} \begin{bmatrix} \mathfrak{a} \\ \mathfrak{b} \end{bmatrix} = 0. \tag{D.7}$$

Similarly, introduce the notation at time instance k as

$$\begin{bmatrix} \mathcal{A} - \mathcal{B} & \mathcal{C} \\ \mathcal{C}^T & \mathcal{D} - \hat{\sigma}_{\tilde{u}}^{k} I \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = 0. \tag{D.8}$$

Let $\hat{\sigma}_{\tilde{u}}^k$ denote the estimate obtained via (2.75). Alternatively, if $\hat{\Sigma}_{\bar{\varphi}_y}^k$ is known, the input measurement noise could be obtained using (2.82) and denote this quantity ς^k. Using perturbation theory for eigenvalues yields

$$\begin{aligned} \varsigma^k &= \lambda_{\min}\left\{B_k(\alpha_k)\right\} = \lambda_{\min}\left\{B_{k-1}(\alpha_{k-1}) + \Delta B_k\right\} \\ &\approx \varsigma^{k-1} + \frac{\mathfrak{b}^T \Delta B_k \mathfrak{b}}{\mathfrak{b}^T \mathfrak{b}}, \end{aligned} \tag{D.9}$$

where the perturbation is given by (cf. (2.83))

$$\begin{aligned} \Delta B_k &= B_k(\alpha_k) - B_{k-1}(\alpha_{k-1}) \\ &= \mathcal{D} - \mathcal{C}^T\left[\mathcal{A} - \mathcal{B}\right]^{-1}\mathcal{C} - \mathfrak{D} + \mathfrak{C}^T\left[\mathfrak{A} - \mathfrak{B}\right]^{-1}\mathfrak{C} \\ &= \mathcal{D} - \mathcal{C}^T\mathcal{F}^{-1}\mathcal{C} - \mathfrak{D} + \mathfrak{C}^T\mathfrak{F}^{-1}\mathfrak{C} \end{aligned} \tag{D.10}$$

with $\mathcal{F} \triangleq [\mathcal{A} - \mathcal{B}]$ and $\mathfrak{F} \triangleq [\mathfrak{A} - \mathfrak{B}]$. Substituting (D.10) in (D.9) yields

$$\begin{aligned}\varsigma^k - \varsigma^{k-1} &\approx \frac{\mathfrak{b}^T}{\mathfrak{b}^T\mathfrak{b}} \left(\mathcal{D} - \mathfrak{D} + \mathfrak{C}^T\mathfrak{F}^{-1}\mathfrak{C} - \mathcal{C}^T\mathcal{F}^{-1}\mathcal{C}\right)\mathfrak{b} \\ &= \frac{\mathfrak{b}^T}{\mathfrak{b}^T\mathfrak{b}} \left(\mathcal{D} - \mathfrak{D}\right)\mathfrak{b} + \frac{\mathfrak{b}^T X \mathfrak{b}}{\mathfrak{b}^T\mathfrak{b}},\end{aligned} \tag{D.11}$$

where X can be expressed as

$$\begin{aligned}X &= \mathfrak{C}^T\mathfrak{F}^{-1}\mathfrak{C} - \mathcal{C}^T\mathcal{F}^{-1}\mathcal{C} \\ &\quad + \mathfrak{C}^T\mathcal{F}^{-1}\mathcal{C} - \mathfrak{C}^T\mathcal{F}^{-1}\mathcal{C} \\ &\quad + \mathfrak{C}^T\mathfrak{F}^{-1}\mathcal{C} - \mathfrak{C}^T\mathfrak{F}^{-1}\mathcal{C} \\ &= \left(\mathfrak{C}^T - \mathcal{C}^T\right)\mathcal{F}^{-1}\mathcal{C} + \mathfrak{C}^T\mathfrak{F}^{-1}\left(\mathfrak{C} - \mathcal{C}\right) \\ &\quad - \mathfrak{C}^T\mathfrak{F}^{-1}\left(\mathfrak{F} - \mathcal{F}\right)\mathcal{F}^{-1}\mathcal{C}\end{aligned} \tag{D.12}$$

and by combining (D.11) and (D.12), it holds that

$$\begin{aligned}\mathfrak{b}^T\mathfrak{b}\left(\varsigma^k - \varsigma^{k-1}\right) &\approx \mathfrak{b}^T\left(\mathcal{D} - \mathfrak{D}\right)\mathfrak{b} \\ &\quad + \mathfrak{b}^T\left(\mathfrak{C}^T - \mathcal{C}^T\right)\mathcal{F}^{-1}\mathcal{C}\mathfrak{b} \\ &\quad + \mathfrak{b}^T\mathfrak{C}^T\mathfrak{F}^{-1}\left(\mathfrak{C} - \mathcal{C}\right)\mathfrak{b} \\ &\quad - \mathfrak{b}^T\mathfrak{C}^T\mathfrak{F}^{-1}\left(\mathfrak{F} - \mathcal{F}\right)\mathcal{F}^{-1}\mathcal{C}\mathfrak{b}.\end{aligned} \tag{D.13}$$

Now, the first row of (D.7) gives

$$\mathfrak{a} = -\mathfrak{F}^{-1}\mathfrak{C}\mathfrak{b} \tag{D.14}$$

and by assuming that $\mathcal{F}^{-1}\mathcal{C}\mathfrak{b} \approx -\mathfrak{a}$, (D.13) finally simplifies to

$$\begin{aligned}\mathfrak{b}^T\mathfrak{b}\left(\varsigma^k - \varsigma^{k-1}\right) &\approx \mathfrak{b}^T\left(\mathcal{D} - \mathfrak{D}\right)\mathfrak{b} \\ &\quad - \mathfrak{b}^T\left(\mathfrak{C}^T - \mathcal{C}^T\right)\mathfrak{a} \\ &\quad - \mathfrak{a}^T\left(\mathfrak{C} - \mathcal{C}\right)\mathfrak{b} \\ &\quad - \mathfrak{a}^T\left(\mathfrak{F} - \mathcal{F}\right)\mathfrak{a},\end{aligned} \tag{D.15}$$

where $\mathcal{F}$ is the only element depending on α_k. Therefore,

$$\frac{d\varsigma^k}{d\alpha_k} \approx \frac{d}{d\alpha_k}\left(\frac{\mathfrak{a}^T\left(\mathcal{A} - \mathcal{B}\right)\mathfrak{a}}{\mathfrak{b}^T\mathfrak{b}}\right) = -\frac{\mathfrak{a}^T\frac{d\mathcal{B}}{d\alpha_k}\mathfrak{a}}{\mathfrak{b}^T\mathfrak{b}} \tag{D.16}$$

or equivalently

$$\frac{d}{d\alpha_k}\lambda_{\min}\left\{B_k(\alpha_k)\right\} \approx -\frac{\hat{\bar{a}}_{k-1}^T}{\hat{b}_{k-1}^T\hat{b}_{k-1}}\frac{d}{d\alpha_k}\hat{\Sigma}_{\hat{\bar{\varphi}}_y}^k\hat{\bar{a}}_{k-1}. \tag{D.17}$$

Since $\Sigma^k_{\bar{\tilde{\varphi}}_y}$ consists of the quantities $\hat{r}^k_{\tilde{y}}(0), ..., \hat{r}^k_{\tilde{y}}(n_a)$, it remains to determine

$$\frac{d}{d\alpha_k}\hat{\rho}^k_y = \begin{bmatrix} \frac{d}{d\alpha_k}\hat{r}^k_{\tilde{y}}(0) & \cdots & \frac{d}{d\alpha_k}\hat{r}^k_{\tilde{y}}(n_a) \end{bmatrix}^T \tag{D.18}$$

which, due to (2.81), is given by

$$\frac{d}{d\alpha_k}\hat{\rho}^k_y = N(H_k). \tag{D.19}$$

Appendix E

Recursive bias compensation of the IV estimator

In order to obtain an explicit expression for the bias, the linear regression formulation

$$y_i = \varphi_i^T \theta + e_i \tag{E.1}$$

is substituted in (4.46) which gives

$$\begin{aligned}\hat{\theta}_k^{\mathrm{IV}} &= \left[\frac{1}{k}\sum_{i=1}^{k}\delta_i^*\varphi_i^T\right]^{-1}\frac{1}{k}\sum_{i=1}^{k}\delta_i^*\left(\varphi_i^T\theta + e_i\right)\\ &= \theta + \left[\frac{1}{k}\sum_{i=1}^{k}\delta_i^*\varphi_i^T\right]^{-1}\frac{1}{k}\sum_{i=1}^{k}\delta_i^* e_i\end{aligned} \tag{E.2}$$

By substituting $e_i = -\tilde{\varphi}_i\theta + \tilde{y}_i$ it follows that

$$\hat{\theta}_k^{\mathrm{IV}} = \theta + \left[\frac{1}{k}\sum_{i=1}^{k}\delta_i\varphi_i^T\right]^{-1}\frac{1}{k}\sum_{i=1}^{k}\delta_i^*\tilde{y}_i - \left[\frac{1}{k}\sum_{i=1}^{k}\delta_i^*\varphi_i^T\right]^{-1}\frac{1}{k}\sum_{i=1}^{k}\delta_i^*\tilde{\varphi}_i^T\theta. \tag{E.3}$$

The vector δ_i^* is uncorrelated with $\tilde{y}_i$ which means that the middle part of the sum in (E.3) diminishes in the asymptotic case, whereas

$$\lim_{k\to\infty}\quad\frac{1}{k}\sum_{i=1}^{k}\begin{bmatrix}\varphi_{u_i}\\ \zeta_i^*\end{bmatrix}\begin{bmatrix}\tilde{\varphi}_{y_i}^T & \tilde{\varphi}_{u_i}^T\end{bmatrix}^T = \begin{bmatrix}0 & \sigma_{\tilde{u}}I_{n_b}\\ 0 & 0\end{bmatrix}, \tag{E.4}$$

where ζ_i^* is obtained by deleting the last entry of ζ_i. Consequently, for $k \to \infty$ (E.3) becomes

$$\theta^{\mathrm{IV}} = \theta - \sigma_{\tilde{u}}\Sigma_{\delta\varphi}^{-1}J^*\theta, \tag{E.5}$$

where J^* is obtained by deleting the last row of J in (2.71). Equation (E.5) gives rise to the recursive bias compensation update equation for $\hat{\theta}_k$

$$\hat{\theta}_k = \hat{\theta}_k^{\mathrm{IV}} + \hat{\sigma}_{\tilde{u}}^k \left[\hat{\Sigma}_{\delta\varphi}^k\right]^{-1} J^* \hat{\theta}_{k-1}, \tag{E.6}$$

where the uncompensated parameter estimate $\hat{\theta}_k^{\mathrm{IV}}$ can be recursively computed via a recursive IV (RIV) algorithm (Ljung 1999, p. 369) given by

$$\hat{\theta}_k^{\mathrm{IV}} = \hat{\theta}_{k-1}^{\mathrm{IV}} + L_k \left[y_k - \varphi_k^T \hat{\theta}_{k-1}^{\mathrm{IV}}\right], \tag{E.7a}$$

$$L_k = \frac{P_{k-1}\delta_k^*}{\frac{1-\gamma_k}{\gamma_k} + \varphi_k^T P_{k-1}\delta_k^*}, \tag{E.7b}$$

$$P_k = \frac{1}{1-\gamma_k}\left[P_{k-1} - \frac{P_{k-1}\delta_k^*\varphi_k^T P_{k-1}}{\frac{1-\gamma_k}{\gamma_k} + \varphi_k^T P_{k-1}\delta_k^*}\right]. \tag{E.7c}$$

with the only difference being that P_k is scaled such that

$$\left[\hat{\Sigma}_{\delta\varphi}^k\right]^{-1} = P_k. \tag{E.8}$$

Appendix F

Left pseudo inverse of $G_k(\hat{\theta}_k)$

Recall that

$$G(\hat{\theta}_k) = \begin{bmatrix} 1 & 0 \\ -\hat{a}_k & 0 \\ 0 & -\hat{b}_k \\ 0 & 0 \end{bmatrix}. \tag{F.1}$$

Then

$$G^T(\hat{\theta}_k)G(\hat{\theta}_k) = \begin{bmatrix} 1 & -\hat{a}_k^T & 0 & 0 \\ 0 & 0 & -\hat{b}_k^T & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\hat{a}_k & 0 \\ 0 & -\hat{b}_k \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 + \hat{a}_k^T\hat{a}_k & 0 \\ 0 & \hat{b}_k^T\hat{b}_k \end{bmatrix}, \tag{F.2}$$

and

$$\left[G^T(\hat{\theta}_k)G(\hat{\theta}_k)\right]^{-1} = \begin{bmatrix} \left(1 + \hat{a}_k^T\hat{a}_k\right)^{-1} & 0 \\ 0 & \left(\hat{b}_k^T\hat{b}_k\right)^{-1} \end{bmatrix}, \tag{F.3}$$

which yields the left pseudo inverse of $G_k(\hat{\theta}_k)$ as

$$\begin{aligned} G^\dagger(\hat{\theta}_k) &= \left[G^T(\hat{\theta}_k)G(\hat{\theta}_k)\right]^{-1} G(\hat{\theta}_k)^T \\ &= \begin{bmatrix} \left(1 + \hat{a}_k^T\hat{a}_k\right)^{-1} & 0 \\ 0 & \left(\hat{b}_k^T\hat{b}_k\right)^{-1} \end{bmatrix} \begin{bmatrix} 1 & -\hat{a}_k^T & 0 & 0 \\ 0 & 0 & -\hat{b}_k^T & 0 \end{bmatrix} \\ &= \begin{bmatrix} \left(1 + \hat{a}_k^T\hat{a}_k\right)^{-1} & -\hat{a}_k^T\left(1 + \hat{a}_k^T\hat{a}_k\right)^{-1} & 0 & 0 \\ 0 & 0 & -\hat{b}_k^T\left(\hat{b}_k^T\hat{b}_k\right)^{-1} & 0 \end{bmatrix}. \end{aligned} \tag{F.4}$$

Appendix G

Recursive algorithms for overdetermined normal equations

G.1 ERLS1

The first algorithm makes use of the matrix inversion lemma for pseudo inverses (Feng et al. 2001) and is denoted ERLS1.

G.1.1 Recursive update of pseudo inverse

Recall that the pseudo matrix inversion lemma is given by (5.24)

$$[A + bc^T]^\dagger = A^\dagger - \frac{A^\dagger b c^T A^\dagger}{1 + c^T A^\dagger b}. \tag{G.1}$$

The update for the weighted arithmetic mean $\hat{\Sigma}_{z\varphi}^k$ is given by (cf. Appendix B)

$$\hat{\Sigma}_{z\varphi}^k = \underbrace{(1-\gamma_k)\hat{\Sigma}_{z\varphi}^{k-1}}_{A} + \underbrace{\gamma_k z_k}_{b}\underbrace{\varphi_k^T}_{c^T}. \tag{G.2}$$

Defining

$$P_k \triangleq \left[\hat{\Sigma}_{z\varphi}^k\right]^\dagger, \tag{G.3}$$

the application of the pseudo matrix inversion lemma yields

$$\begin{aligned} P_k &= \frac{1}{1-\gamma_k} P_{k-1} - \frac{\frac{1}{1-\gamma_k} P_{k-1}\gamma_k z_k \varphi_k^T \frac{1}{1-\gamma_k} P_{k-1}}{1 + \varphi_k^T \frac{1}{1-\gamma_k} P_{k-1}\gamma_k z_k} \\ &= \frac{1}{1-\gamma_k}\left(P_{k-1} - \frac{P_{k-1}\gamma_k z_k \varphi_k^T P_{k-1}}{1-\gamma_k + \varphi_k^T P_{k-1}\gamma_k z_k}\right) \\ &= \frac{1}{1-\gamma_k}\left(P_{k-1} - L_k \varphi_k^T P_{k-1}\right), \end{aligned} \tag{G.4}$$

where

$$L_k \triangleq \frac{P_{k-1}\gamma_k z_k}{1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k}. \tag{G.5}$$

G.1.2 Recursive update of θ_k^{IV}

The recursive update of the parameter vector is given by

$$\begin{aligned}
\theta_k^{\mathrm{IV}} &= P_k \hat{\xi}_{zy}^k \\
&= \frac{1}{1-\gamma_k}\left(P_{k-1} - L_k\varphi_k^T P_{k-1}\right)\left[(1-\gamma_k)\hat{\xi}_{z\varphi}^{k-1} + \gamma_k z_k y_k\right] \\
&= \theta_{k-1}^{\mathrm{IV}} - L_k\varphi_k^T\theta_{k-1}^{\mathrm{IV}} + \frac{\gamma_k}{1-\gamma_k}P_{k-1}z_k y_k - \frac{\gamma_k}{1-\gamma_k}L_k\varphi_k^T P_{k-1}z_k y_k \\
&= \theta_{k-1}^{\mathrm{IV}} - L_k\varphi_k^T\theta_{k-1}^{\mathrm{IV}} + \left(\frac{\gamma_k}{1-\gamma_k}P_{k-1}z_k - \frac{\gamma_k}{1-\gamma_k}L_k\varphi_k^T P_{k-1}z_k\right)y_k,
\end{aligned} \tag{G.6}$$

and since

$$\begin{aligned}
&\frac{\gamma_k}{1-\gamma_k}P_{k-1}z_k - \frac{\gamma_k}{1-\gamma_k}L_k\varphi_k^T P_{k-1}z_k \\
=&\frac{\gamma_k}{1-\gamma_k}\left(P_{k-1}z_k - \frac{P_{k-1}\gamma_k z_k\varphi_k^T P_{k-1}z_k}{1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k}\right) \\
=&\frac{\gamma_k}{1-\gamma_k}\left(\frac{P_{k-1}z_k\left(1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k\right)}{1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k} - \frac{P_{k-1}\gamma_k z_k\varphi_k^T P_{k-1}z_k}{1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k}\right) \\
=&\frac{\gamma_k}{1-\gamma_k}\left(\frac{P_{k-1}z_k\left(1-\gamma_k\right)}{1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k}\right) \\
=&\frac{1}{1-\gamma_k}\left(\frac{\gamma_k P_{k-1}z_k}{1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k} - \frac{\gamma_k P_{k-1}z_k\gamma_k}{1-\gamma_k+\varphi_k^T P_{k-1}\gamma_k z_k}\right) \\
=&\frac{1}{1-\gamma_k}\left(L_k - \gamma L_k\right) \\
=&\frac{1-\gamma_k}{1-\gamma_k}L_k \\
=&L_k,
\end{aligned} \tag{G.7}$$

(G.6) simplifies to

$$\begin{aligned}
\theta_k^{\mathrm{IV}} &= \theta_{k-1}^{\mathrm{IV}} - L_k\varphi_k^T\theta_{k-1}^{\mathrm{IV}} + L_k y_k \\
&= \theta_{k-1}^{\mathrm{IV}} + L_k\left(y_k - \varphi_k^T\theta_{k-1}^{\mathrm{IV}}\right).
\end{aligned} \tag{G.8}$$

The extended recursive least squares algorithm, denoted ERLS1, can be summarised as follows.

Algorithm G.1 (ERLS1).

$$\theta_k^{\mathrm{IV}} = \theta_{k-1}^{\mathrm{IV}} + L_k \left(y_k - \varphi_k^T \theta_{k-1}^{\mathrm{IV}} \right) \tag{G.9a}$$

$$L_k = \frac{P_{k-1}\gamma_k z_k}{1 - \gamma_k + \varphi_k^T P_{k-1} \gamma_k z_k} \tag{G.9b}$$

$$P_k = \frac{1}{1-\gamma_k} \left(P_{k-1} - L_k \varphi_k^T P_{k-1} \right) \tag{G.9c}$$

G.2 ERLS2

The second algorithm uses the standard matrix inversion lemma applied to a reformulated problem, which reduces the dimension of matrix to be inverted to a fixed size of 2×2. This method is given in (Söderström & Stoica 1989).

Consider the overdetermined system of normal equations

$$\Sigma_{z\varphi}^k \theta = \xi_{zy}^k, \tag{G.10}$$

for which the least squares solution is given by

$$\hat{\theta}_k = \Sigma_{z\varphi}^{k\dagger} \xi_{zy}^k = \left[\Sigma_{z\varphi}^{k^T} \Sigma_{z\varphi}^k \right]^{-1} \Sigma_{z\varphi}^{k^T} \xi_{zy}^k = P_k \Sigma_{z\varphi}^{k^T} \xi_{zy}^k. \tag{G.11}$$

where

$$P_k \triangleq \left[\Sigma_{z\varphi}^{k^T} \Sigma_{z\varphi}^k \right]^{-1}. \tag{G.12}$$

A recursive expression for $\hat{\theta}_k$ is then given by

$$\hat{\theta}_k = \hat{\theta}_{k-1} + P_k \Sigma_{z\varphi}^{k^T} \left[\xi_{zy}^k - \Sigma_{z\varphi}^k \hat{\theta}_{k-1} \right]. \tag{G.13}$$

It follows that

$$\begin{aligned} \Sigma_{z\varphi}^{k^T} \left[\xi_{zy}^k - \Sigma_{z\varphi}^k \hat{\theta}_{k-1} \right] &= \left[\Sigma_{z\varphi}^{k-1^T} + \varphi_k z_k^T \right] \left[\xi_{zy}^{k-1} + z_k y_k - \left(\Sigma_{z\varphi}^{k-1} + z_k \varphi_k^T \right) \hat{\theta}_{k-1} \right] \\ &= \Sigma_{z\varphi}^{k-1^T} \left(\xi_{zy}^{k-1} - \Sigma_{z\varphi}^{k-1} \hat{\theta}_{k-1} \right) + \Sigma_{z\varphi}^{k-1^T} z_k \left(y_k - \varphi_k^T \hat{\theta}_{k-1} \right) \\ &\quad + \varphi_k z_k^T \left[\xi_{zy}^{k-1} - \Sigma_{z\varphi}^{k-1} \hat{\theta}_{k-1} + z_k \left(y_k - \varphi_k^T \hat{\theta}_{k-1} \right) \right]. \end{aligned} \tag{G.14}$$

Noting that $\xi_{zy}^{k-1} - \Sigma_{z\varphi}^{k-1}\hat{\theta}_{k-1} = 0$, (G.14) can be re-expressed as

$$\begin{aligned}\Sigma_{z\varphi}^{k^T}\left[\xi_{zy}^{k} - \Sigma_{z\varphi}^{k}\hat{\theta}_{k-1}\right] &= \left[\varphi_k, \quad \Sigma_{z\varphi}^{k-1^T} z_k + \varphi_k z_k^T z_k\right]\begin{bmatrix} z_k^T\left(\xi_{zy}^{k-1} - \Sigma_{z\varphi}^{k-1}\hat{\theta}_{k-1}\right) \\ y_k - \varphi_k^T\hat{\theta}_{k-1}\end{bmatrix} \\ &= \left[\Sigma_{z\varphi}^{k-1^T} z_k, \varphi_k\right]\begin{bmatrix} 0 & 1 \\ 1 z_k^T z_k & \end{bmatrix}\left(\begin{bmatrix} z_k^T\xi_{zy}^{k-1} \\ y_k\end{bmatrix} - \begin{bmatrix} w_k^T \\ \varphi_k^t\end{bmatrix}\hat{\theta}_{k-1}\right) \\ &= \phi\Lambda_k^{-1}\left(v_k - \phi_k^T\hat{\theta}_{k-1}\right), \end{aligned} \tag{G.15}$$

where

$$w_k \triangleq \Sigma_{z\varphi}^{k-1^T} z_k, \tag{G.16a}$$

$$\phi_k \triangleq \begin{bmatrix} w_k & \varphi_k\end{bmatrix}, \tag{G.16b}$$

$$\Lambda_k^{-1} \triangleq \begin{bmatrix} 0 & 1 \\ 1 & z_k^T z_k\end{bmatrix}, \tag{G.16c}$$

$$v_k \triangleq \begin{bmatrix} z_k^T\xi_{zy}^{k-1} \\ y_k\end{bmatrix}. \tag{G.16d}$$

Using (G.15), it holds

$$\begin{aligned} P_k^{-1} &= \left(\Sigma_{z\varphi}^{k-1^T} + \varphi_k z_k^T\right)\left(\Sigma_{z\varphi}^{k-1} + z_k\varphi_k^T\right) \\ &= P_{k-1}^{-1} + \varphi_k w_k^T + w_k\varphi_k^T + \varphi_k z_k^T z_k\varphi_k^T \\ &= P_{k-1}^{-1} + \begin{bmatrix} w_k & \varphi_k\end{bmatrix}\begin{bmatrix} 0 & 1 \\ 1 & z_k^T z_k\end{bmatrix}\begin{bmatrix} w_k^T \\ \varphi_k^T\end{bmatrix} \\ &= P_{k-1}^{-1} + \phi\Lambda_k^{-1}\phi_k^T, \end{aligned} \tag{G.17}$$

which allows an application of the (standard) matrix inversion lemma giving

$$P_k = P_{k-1} - P_{k-1}\phi_k\left(\Lambda_k + \phi_k^T P_{k-1}\phi_k\right)^{-1}\phi_k^T P_{k-1}, \tag{G.18}$$

and

$$P_k\phi_k = P_{k-1}\phi_k\left(\Lambda_k + \phi_k^T P_{k-1}\phi_k\right)^{-1}\Lambda_k. \tag{G.19}$$

The ERLS2 algorithm can be summarised as follows.

Algorithm G.2 (ERLS2).

$$\hat{\theta}_k = \hat{\theta}_{k-1} + L_k \left(v_k - \phi \hat{\theta}_{k-1} \right) \tag{G.20a}$$

$$L_k = P_{k-1}\phi_k \left(\Lambda_k + \phi_k^T P_{k-1} \phi_k \right)^{-1} \tag{G.20b}$$

$$P_k = P_{k-1} - L_k \phi_k^T P_{k-1} \tag{G.20c}$$

$$w_k = \Sigma_{z\varphi}^{k-1^T} z_k \tag{G.20d}$$

$$\phi_k = \begin{bmatrix} w_k & \varphi_k \end{bmatrix} \tag{G.20e}$$

$$\Lambda_k = \begin{bmatrix} -z_k^T z_k & 1 \\ 1 & 0 \end{bmatrix} \tag{G.20f}$$

$$v_k = \begin{bmatrix} z_k^T \xi_{zy}^{k-1} \\ y_k \end{bmatrix} \tag{G.20g}$$

Appendix H

Derivation of the Kalman filter for bilinear systems

The single phase Kalman filter (one-step prediction problem giving $\hat{x}_{k+1|k}$) is derived for a bilinear discrete-time system based on an innovations approach. As outlined in (Favoreel et al. 1999), a bilinear system can be regarded as a time-varying linear system, hence, the derivation follows the approach given in (Anderson & Moore 1979, Sec. 5.4).

H.1 Preliminaries

For $k \geq 0$, let the bilinear system be given by

$$x_{k+1} = A_k x_k + B_k u_k + N_k u_k x_k + G_k v_k, \tag{H.1a}$$

$$z_k = C_k x_k + D_k u_k + e_k, \tag{H.1b}$$

where x_k is the state vector, u_k the system input, z_k the measured system output and v_k and e_k are state noise and output noise, respectively. The system matrices A_k, B_k, C_k, D_k, G_k and N_k depend on the time instance k and are of appropriate dimensions. Additionally, the following assumptions hold:

AI2 The system input u_k is known exactly.

AN3 The noise sequences v_k and e_k are zero mean, white, and satisfy

$$E\left[\begin{bmatrix} v_k \\ e_k \end{bmatrix}\begin{bmatrix} v_l^T & e_l^T \end{bmatrix}\right] = \begin{bmatrix} \Sigma_v^k & \Sigma_{ve}^k \\ \Sigma_{ve}^{k^T} & \Sigma_e^k \end{bmatrix}\delta_{kl}. \tag{H.2}$$

AN4 The initial state x_0 has the mean $\bar{x}_0$ with covariance matrix P_0. In addition, x_0 is independent of $\begin{bmatrix} v_k^T & e_k^T \end{bmatrix}^T$ for all k.

AN5 The quantities x_0, v_k and e_k are jointly Gaussian.

Basically, there are two different approaches to tackle the problem: The first one is to exploit Assumption **AN5** and derive the optimal minimum variance (conditional mean) estimator $\hat{x}_{k|k-1} = E[x_k|Z_{k-1}]$, where $E[\cdot]$ denotes the expected value operator and $Z_k = \{z_0, z_1, ..., z_k\}$. In this case, the corresponding covariance matrix $P_{k|k-1}$ is both, conditional and unconditional. Alternatively, one can drop **AN5** and seek the linear minimum variance estimator $\hat{x}_{k|k-1} = E^*[x_k|Z_{k-1}]$, (which is not an expectation) where the associated covariance matrix $P_{k|k-1}$ is unconditional (cf. (Anderson & Moore 1979, Sec. 5.2) for more details). Here, the first approach is followed.

H.2 Evolution of the conditional mean $\hat{x}_{k|k-1}$

The objective is to find

$$\hat{x}_{k|k-1} = E[x_k|Z_{k-1}] = E[x_k|\tilde{Z}_{k-1}], \tag{H.3}$$

where $\tilde{Z}_{k-1}$ is the sequence of innovations $\tilde{z}_0, \tilde{z}_1, ..., \tilde{z}_{k-1}$, with

$$\begin{aligned}\tilde{z}_k &= z_k - E[z_k|Z_{k-1}] = z_k - E[z_k|\tilde{Z}_{k-1}] \\ &= z_k - C_k\hat{x}_{k|k-1} - D_k u_k \\ &= C_k x_k + D_k u_k + e_k - C_k\hat{x}_{k|k-1} - D_k u_k \\ &= C_k\tilde{x}_k + e_k,\end{aligned} \tag{H.4}$$

and

$$\tilde{x}_k = x_k - \hat{x}_{k|k-1} \tag{H.5}$$

denotes the state prediction error with associated error covariance matrix

$$P_{k|k-1} = E[\tilde{x}_k\tilde{x}_k^T]. \tag{H.6}$$

Since the innovations are independent, one can write for (H.3) (Anderson & Moore 1979, Theorem 2.4, p. 94)

$$\hat{x}_{k+1|k} = E[x_{k+1}|\tilde{z}_0, ..., \tilde{z}_k] = E[x_{k+1}|\tilde{z}_k] + E[x_{k+1}|\tilde{z}_0, ..., \tilde{z}_{k-1}] - E[x_{k+1}], \tag{H.7}$$

and due to the fact that x_{k+1} and $\tilde{z}_k$ are jointly Gaussian, one can write for the first term (Anderson & Moore 1979, Theorem 2.1, p. 93)

$$E[x_{k+1}|\tilde{z}_k] = E[x_{k+1}] + \mathrm{cov}(x_{k+1}, \tilde{z}_k)\left[\mathrm{cov}(\tilde{z}_k, \tilde{z}_k)\right]^{-1}\tilde{z}_k. \tag{H.8}$$

Restricting attention the the first covariance term in (H.8), one obtains utilising

(H.4)

$$\begin{aligned}\operatorname{cov}(x_{k+1},\tilde{z}_k) &= E\left[x_k\tilde{z}_k^T\right]\\ &= E\left[\left[A_kx_k+B_ku_k+N_ku_kx_k+G_kv_k\right]\left[C_k\tilde{x}_k+e_k\right]^T\right].\end{aligned} \tag{H.9}$$

By exploiting the independence of the various variables and since u_k is known and $\tilde{x}_k$, hence $C_k\tilde{x}_k$, has zero mean, one obtains

$$\begin{aligned}\operatorname{cov}(x_{k+1},\tilde{z}_k) &= E\left[A_kx_k\tilde{x}_k^TC_k^T\right]+E\left[N_ku_kx_k\tilde{x}_k^TC_k^T\right]+E\left[G_kv_ke_k^T\right]\\ &= A_kE\left[\tilde{x}_k\tilde{x}_k^T\right]C_k^T+A_kE\left[\hat{x}_{k|k-1}\tilde{x}_k^T\right]C_k^T\\ &\quad+N_ku_kE\left[\tilde{x}_k\tilde{x}_k^T\right]C_k^T+N_ku_kE\left[\hat{x}_{k|k-1}\tilde{x}_k^T\right]C_k^T+G_k\Sigma_{ve}^k.\end{aligned} \tag{H.10}$$

The quantity $E\left[\hat{x}_{k|k-1}\tilde{x}_k^T\right]$ is equal to zero, since "$\tilde{x}_k$ is the error in projecting x_k onto the subspace generated by $z_{k-1}, z_{k-2}, ...$ and is, therefore, orthogonal to that subspace, while $\hat{x}_{k|k-1}$ is a member of it" (Anderson & Moore 1979, p. 106). This gives

$$\operatorname{cov}(x_{k+1},\tilde{z}_k) = A_kP_{k|k-1}C_k^T+N_ku_kP_{k|k-1}C_k^T+G_k\Sigma_{ve}^k. \tag{H.11}$$

The second covariance term in (H.8) is given by

$$\begin{aligned}\operatorname{cov}(\tilde{z}_k,\tilde{z}_k) &= E\left[\tilde{z}_k\tilde{z}_k^T\right]\\ &= E\left[C_k\tilde{x}_k+e_k\right]\left[C_k\tilde{x}_k+e_k\right]^T\\ &= C_kP_{k|k-1}C_k^T+\Sigma_e^k.\end{aligned} \tag{H.12}$$

Therefore, (H.8) becomes

$$\begin{aligned}E[x_{k+1}|\tilde{z}_k] = E[x_{k+1}] &+ \left[A_kP_{k|k-1}C_k^T+N_ku_kP_{k|k-1}C_k^T+G_k\Sigma_{ve}^k\right]\\ &\times\left[C_kP_{k|k-1}C_k^T+\Sigma_e^k\right]^{-1}\tilde{z}_k.\end{aligned} \tag{H.13}$$

In order to determine the prediction of the state, it remains to determine the second term in (H.7), which is given by

$$E[x_{k+1}|\tilde{Z}_{k-1}] = E[A_kx_k+B_ku_k+N_ku_kx_k+G_kv_k|\tilde{Z}_{k-1}], \tag{H.14}$$

and due to the independence of v_k and $\tilde{Z}_{k-1}$ and the fact that u_k is known, one obtains

$$E[x_{k+1}|\tilde{Z}_{k-1}] = A_k\hat{x}_{k|k-1}+B_ku_k+N_ku_k\hat{x}_{k|k-1}. \tag{H.15}$$

Hence, the state prediction of (H.7) becomes

$$\begin{aligned}\hat{x}_{k+1|k} = E[x_{k+1}] &+ \left[A_k P_{k|k-1} C_k^T + N_k u_k P_{k|k-1} C_k^T + G_k \Sigma_{ve}^k\right] \left[C_k P_{k|k-1} C_k^T + \Sigma_e^k\right]^{-1} \tilde{z}_k \\ &+ A_k \hat{x}_{k|k-1} + B_k u_k + N_k u_k \hat{x}_{k|k-1} - E[x_{k+1}], \end{aligned} \tag{H.16}$$

which is equivalent to

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k-1} + B_k u_k + N_k u_k \hat{x}_{k|k-1} + K_k \left[z_k - C_k \hat{x}_{k|k-1} - D_k u_k\right], \tag{H.17a}$$

$$K_k = \left[A_k P_{k|k-1} C_k^T + N_k u_k P_{k|k-1} C_k^T + G_k \Sigma_{ve}^k\right] \left[C_k P_{k|k-1} C_k^T + \Sigma_e^k\right]^{-1}. \tag{H.17b}$$

H.3 Evolution of the covariance matrix

The state estimation error is given by

$$\begin{aligned}\tilde{x}_{k+1} &= x_{k+1} - \hat{x}_{k+1|k} \\ &= A_k x_k + B_k u_k + N_k u_k x_k + G_k v_k - A_k \hat{x}_{k|k-1} - B_k u_k - N_k u_k \hat{x}_{k|k-1} - K_k \tilde{z}_k \\ &= A_k \tilde{x}_k + N_k u_k \tilde{x}_k + G_k v_k - K_k \left[C_k \tilde{x}_k + e_k\right] \\ &= \left[A_k + N_k u_k - K_k C_k\right] \tilde{x}_k + G_k v_k - K_k e_k, \end{aligned} \tag{H.18}$$

and consequently, since $\tilde{x}_k$, v_k and e_k are mutually independent, the corresponding covariance matrix is

$$\begin{aligned}P_{k+1|k} &= E\left[\tilde{x}_{k+1} \tilde{x}_{k+1}^T\right] \\ &= \left[A_k + N_k u_k - K_k C_k\right] P_{k|k-1} \left[A_k^T + u_k^T N_k^T - C_k^T K_k^T\right] \\ &\quad + G_k \Sigma_v^k G_k^T - G_k \Sigma_{ve}^k K_k^T - K_k {\Sigma_{ve}^k}^T G_k^T + K_k \Sigma_e^k K_k^T \\ &= \left[A_k + N_k u_k - K_k C_k\right] P_{k|k-1} \left[A_k^T + u_k^T N_k^T - C_k^T K_k^T\right] \\ &\quad + \begin{bmatrix} G_k & -K_k \end{bmatrix} \begin{bmatrix} \Sigma_v^k & \Sigma_{ve}^k \\ {\Sigma_{ve}^k}^T & \Sigma_e^k \end{bmatrix} \begin{bmatrix} G_k^T \\ -K_k^T \end{bmatrix}. \end{aligned} \tag{H.19}$$

Alternative formulation In the literature, the Kalman filter covariance matrix frequently occurs in a slightly different form, which is obtained by replacing the Kalman gain K_k into (H.19). This gives

$$\begin{aligned}P_{k+1|k} &= \left[A_k + N_k u_k - K_k C_k\right] P_{k|k-1} \left[A_k^T + u_k^T N_k^T - C_k^T K_k^T\right] \\ &\quad + G_k \Sigma_v^k G_k^T - G_k \Sigma_{ve}^k K_k^T - K_k {\Sigma_{ve}^k}^T G_k^T + K_k \Sigma_e^k K_k^T \\ &= A_k P_{k|k-1} A_k^T + A_k P_{k|k-1} u_k^T N_k^T - A_k P_{k|k-1} C_k^T K_k^T \\ &\quad + N_k u_k P_{k|k-1} A_k^T + N_k u_k P_{k|k-1} u_k^T N_k^T - N_k u_k P_{k|k-1} C_k^T K_k^T \\ &\quad - K_k C_k P_{k|k-1} A_k^T - K_k C_k P_{k|k-1} u_k^T N_k^T + K_k C_k P_{k|k-1} C_k^T K_k^T \\ &\quad + G_k \Sigma_v^k G_k^T - G_k \Sigma_{ve}^k K_k^T - K_k {\Sigma_{ve}^k}^T G_k^T + K_k \Sigma_e^k K_k^T \end{aligned}$$

$$
\begin{aligned}
&= A_k P_{k|k-1} A_k^T + N_k u_k P_{k|k-1} u_k^T N_k^T + G_k \Sigma_v^k G_k^T \\
&\quad - \left[A_k P_{k|k-1} C_k^T + N_k u_k P_{k|k-1} C_k^T + G_k \Sigma_{ve}^k \right] K_k^T \\
&\quad - K_k \left[C_k P_{k|k-1} A_k^T + C_k P_{k|k-1} u_k^T N_k^T + {\Sigma_{ve}^k}^T G_k^T \right] \\
&\quad + K_k \left[C_k P_{k|k-1} C_k^T + \Sigma_e^k \right] K_k^T . \qquad \text{(H.20)}
\end{aligned}
$$

Notice, that by utilising the Kalman gain equation (H.17b), it is seen that the last three terms of the sum in (H.20) are, apart form their sign, identical to

$$
\begin{aligned}
&\left[A_k P_{k|k-1} C_k^T + N_k u_k P_{k|k-1} C_k^T + G_k \Sigma_{ve}^k \right] \left[C_k P_{k|k-1} C_k^T + \Sigma_e^k \right]^{-1} \\
&\quad \times \left[C_k P_{k|k-1} A_k^T + C_k P_{k|k-1} u_k^T N_k^T + {\Sigma_{ve}^k}^T G_k^T \right] , \qquad \text{(H.21)}
\end{aligned}
$$

hence, (H.20) simplifies to

$$
\begin{aligned}
P_{k+1|k} &= A_k P_{k|k-1} A_k^T + N_k u_k P_{k|k-1} u_k^T N_k^T + G_k \Sigma_v^k G_k^T \\
&\quad - K_k \left[C_k P_{k|k-1} C_k^T + \Sigma_e^k \right] K_k^T . \qquad \text{(H.22)}
\end{aligned}
$$

H.4 Summary

The single-phase Kalman filter for the bilinear case (BKF) is given by the following algorithm.

Algorithm H.1 (BKF).

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k-1} + B_k u_k + N_k u_k \hat{x}_{k|k-1} + K_k \left[z_k - C_k \hat{x}_{k|k-1} - D_k u_k \right] \qquad \text{(H.23a)}$$

$$K_k = \left[A_k P_{k|k-1} C_k^T + N_k u_k P_{k|k-1} C_k^T + G_k \Sigma_{ve}^k \right] \left[C_k P_{k|k-1} C_k^T + \Sigma_e^k \right]^{-1} \qquad \text{(H.23b)}$$

$$
\begin{aligned}
P_{k+1|k} &= A_k P_{k|k-1} A_k^T + N_k u_k P_{k|k-1} u_k^T N_k^T + G_k \Sigma_v^k G_k^T \\
&\quad - K_k \left[C_k P_{k|k-1} C_k^T + \Sigma_e^k \right] K_k^T \qquad \text{(H.23c)}
\end{aligned}
$$

Note, that this is identical to the standard Kalman filter where the transition matrix is replaced by $\bar{A}_k = A_k + N_k u_k$.

Appendix I

Actual estimation error for the BEIVKF3

Defining the error in the input estimate as

$$\begin{aligned}\tilde{u}_{0_k} &\triangleq u_{0_k} - \hat{u}_{0_k} \\ &= -\tilde{u}_k + \left[\sigma_{\tilde{u}\tilde{y}} - \sigma_{\tilde{u}}\mathcal{D}^T\right]\left[\Sigma_\varepsilon^k\right]^{-1}\left[z_k - \mathcal{C}\hat{x}_{k|k-1}\right] \\ &= -\tilde{u}_k + H_u\left[\Sigma_\varepsilon^k\right]^{-1}\left[\mathcal{C}\tilde{x}_k + e_k\right], \end{aligned} \tag{I.1}$$

where

$$H_u \triangleq \sigma_{\tilde{u}\tilde{y}} - \sigma_{\tilde{u}}\mathcal{D}^T \tag{I.2}$$

is used for ease of notation. Using $\hat{u}_{0_k} = u_{0_k} - \tilde{u}_{0_k}$, the actual state estimation error is given by

$$\begin{aligned}\tilde{x}_{k+1} &= x_{k+1} - \hat{x}_{k+1|k} \\ &= \mathcal{A}_k^a x_k + \mathcal{B}u_k + v_k - \left[\hat{\mathcal{A}}_k\hat{x}_{k|k-1} + \mathcal{B}u_k + \bar{K}_k\left[z_k - \mathcal{C}\hat{x}_{k|k-1}\right]\right] \\ &= \mathcal{A}_k^a x_k + \mathcal{G}w_k - \mathcal{B}\tilde{u}_k - \mathcal{A}_k^a\hat{x}_{k|k-1} - \mathcal{N}\tilde{u}_{0_k}\hat{x}_{k|k-1} - \bar{K}_k\left[\mathcal{C}\tilde{x}_k + e_k\right] \\ &= \left[\mathcal{A} - \bar{K}_k\mathcal{C}\right]\tilde{x}_k + \mathcal{N}u_{0_k}\tilde{x}_k + \mathcal{G}w_k - \bar{K}_k\tilde{y}_k + \left[\bar{K}_k\mathcal{D} - \mathcal{B}\right]\tilde{u}_k - \mathcal{N}\tilde{u}_{0_k}\hat{x}_{k|k-1}, \end{aligned} \tag{I.3}$$

where

$$\begin{aligned}
\mathcal{N}\tilde{u}_{0_k}\hat{x}_{k|k-1} &= \mathcal{N}\left[-\tilde{u}_k + H_u\left[\Sigma_\varepsilon^k\right]^{-1}\left[\mathcal{C}\tilde{x}_k + e_k\right]\right]\left[x_k - \tilde{x}_k\right] \\
&= \left[-\mathcal{N}\tilde{u}_k + \mathcal{N}H_u\left[\Sigma_\varepsilon^k\right]^{-1}\mathcal{C}\tilde{x}_k + \mathcal{N}H_u\left[\Sigma_\varepsilon^k\right]^{-1}e_k\right]\left[x_k - \tilde{x}_k\right] \\
&= -\mathcal{N}\tilde{u}_k x_k + \mathcal{N}H_u\left[\Sigma_\varepsilon^k\right]^{-1}\mathcal{C}\tilde{x}_k x_k + \mathcal{N}H_u\left[\Sigma_\varepsilon^k\right]^{-1}e_k x_k \\
&\quad + \mathcal{N}\tilde{u}_k\tilde{x}_k - \mathcal{N}H_u\left[\Sigma_\varepsilon^k\right]^{-1}\mathcal{C}\tilde{x}_k\tilde{x}_k - \mathcal{N}H_u\left[\Sigma_\varepsilon^k\right]^{-1}e_k\tilde{x}_k. \qquad \text{(I.4)}
\end{aligned}$$

Appendix J

Derivation of the JEKF

This appendix reviews the development of the extended Kalman filter for joint state and parameter estimation given in (Ljung 1979). First, the extended Kalman filter is reviewed followed by its application for joint parameter estimation. Finally, by introducing appropriate block matrices, the expression given in (2.117) is obtained.

J.1 Extended Kalman filter

The extended Kalman filter (EKF) is probably the most widely used tool to estimate the states of a nonlinear state space system. The key idea is to linearise the state space equations at each time instance and to apply linear Kalman filter theory.

Let the discrete-time nonlinear state space system be given by

$$\eta_{k+1} = q_k(\eta_k, u_k) + \bar{v}_k(\eta_k), \tag{J.1a}$$

$$z_k = r_k(\eta_k, u_k) + e_k(\eta_k), \tag{J.1b}$$

where η_k is the state vector, z_k the system output, and where q_k and r_k denote nonlinear functions, whilst process noise and output noise are denoted $\bar{v}_k(\eta_k)$ and $e_k(\eta_k)$, respectively. Note that, in contrast to the scenario considered in (Ljung 1979), here the noise sequences are functions of the unknown state. This generalisation is introduced for the application to the EIV case, where the process and output noise are, depending on the chosen parametrisation, functions of the parameter vector to be estimated. One possibility to estimate the state based on the observations up to time k, which is denoted $\hat{\eta}_{k+1}$, is the EKF which is summarised as follows (Ljung 1979).

Algorithm J.1 (EKF).

$$\hat{\eta}_{k+1} = q_k\left(\hat{\eta}_k, u_k\right) + N_k\left[z_k - r_k\left(\hat{\eta}_k, u_k\right)\right] \tag{J.2a}$$

$$\begin{aligned} N_k = &\left[Q_k(\hat{\eta}_k, u_k) P_k R_k^T(\hat{\eta}_k, u_k) + \Sigma_{\bar{v}e}(\hat{\eta}_k)\right] \\ &\times \left[R_k(\hat{\eta}_k, u_k) P_k R_k^T(\hat{\eta}_k, u_k) + \Sigma_e(\hat{\eta}_k)\right]^{-1} \end{aligned} \tag{J.2b}$$

$$\begin{aligned} P_{k+1} = &Q_k(\hat{\eta}_k, u_k) P_k Q_k^T(\hat{\eta}_k, u_k) + \Sigma_{\bar{v}}(\hat{\eta}_k) \\ &- N_k\left[\Sigma_e(\hat{\eta}_k) + R_k(\hat{\eta}_k, u_k) P_k R_k^T(\hat{\eta}_k, u_k)\right] N_k^T \end{aligned} \tag{J.2c}$$

The Jacobians in Algorithm J.1 are given by

$$Q_k(\hat{\eta}_k, u_k) = \left.\frac{\partial}{\partial \eta} q_k(\eta_k, u_k)\right|_{\eta=\hat{\eta}_k}, \tag{J.3a}$$

$$R_k(\hat{\eta}_k, u_k) = \left.\frac{\partial}{\partial \eta} r_k(\eta_k, u_k)\right|_{\eta=\hat{\eta}_k}, \tag{J.3b}$$

whilst the covariance matrices are defined by

$$\Sigma_{\bar{v}}(\eta_k) = E\left[\bar{v}_k(\eta_k)\bar{v}_l^T(\eta_k)\right]\delta_{kl}, \tag{J.4}$$

$$\Sigma_e(\eta_k) = E\left[e_k(\eta_k)e_l^T(\eta_k)\right]\delta_{kl}, \tag{J.5}$$

$$\Sigma_{\bar{v}e}(\eta_k) = E\left[\bar{v}_k(\eta_k)e_l^T(\eta_k)\right]\delta_{kl}. \tag{J.6}$$

J.2 Application of the EKF for joint state and parameter estimation

Consider the nonlinear system (2.114) which can be brought into the form of (J.1) by defining

$$\eta_k = \begin{bmatrix} x_k \\ \theta_k \end{bmatrix}, \tag{J.7a}$$

$$q_k(\eta_k, u_k) = \begin{bmatrix} A(\theta_k)x_k + B(\theta_k)u_k \\ \theta_k \end{bmatrix}, \tag{J.7b}$$

$$r_k(\eta_k, u_k) = C(\theta_k)x_k + D(\theta_k)u_k, \tag{J.7c}$$

$$\bar{v}_k = \begin{bmatrix} v_k \\ d_k \end{bmatrix}. \tag{J.7d}$$

Hence, the EKF given by (J.2) can be applied to estimate the states and parameters of (2.114). The Jacobians for this case become

$$Q_k(\hat{\eta}_k, u_k) = \begin{bmatrix} A_k & F_k \\ 0 & I \end{bmatrix}, \tag{J.8a}$$

$$R_k(\hat{\eta}_k, u_k) = \begin{bmatrix} C_k & H_k \end{bmatrix}, \tag{J.8b}$$

where

$$F_k = F(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k), \tag{J.9a}$$

$$H_k = H(\hat{\theta}_k, \hat{x}_{k|k-1}, u_k), \tag{J.9b}$$

with F and H being defined by (2.119)

$$F(\hat{\theta}, x, u) = \frac{\partial}{\partial \theta} \left[A(\theta)x + B(\theta)u \right] \big|_{\theta=\hat{\theta}}, \tag{J.10a}$$

$$H(\hat{\theta}, x, u) = \frac{\partial}{\partial \theta} \left[C(\theta)x + D(\theta)u \right] \big|_{\theta=\hat{\theta}}. \tag{J.10b}$$

The noise covariance matrices are given by

$$\Sigma_{\bar{v}} = \begin{bmatrix} \Sigma_v & 0 \\ 0 & \Sigma_d \end{bmatrix}, \tag{J.11a}$$

$$\Sigma_{\bar{v}e} = \begin{bmatrix} \Sigma_{ve} \\ 0 \end{bmatrix}, \tag{J.11b}$$

whereas the initial state and its corresponding error covariance matrix are given by

$$\hat{\eta}_0 = \begin{bmatrix} 0 \\ \hat{\theta}_0 \end{bmatrix}, \tag{J.12a}$$

$$P_0 = \begin{bmatrix} P_{1_0} & 0 \\ 0 & P_{3_0} \end{bmatrix}. \tag{J.12b}$$

The quantities $\hat{\theta}_0$ and P_{3_0} represent a priori knowledge of the parameter estimates, see (Sayed & Kailath 1994*a*, Sayed & Kailath 1994*b*) for a detailed treatment.

J.3 Block form

Introducing the natural block structure

$$N_k \triangleq \begin{bmatrix} K_k \\ L_k \end{bmatrix}, \tag{J.13a}$$

$$P_k \triangleq \begin{bmatrix} P_{1_k} & P_{2_k} \\ P_{2_k}^T & P_{3_k} \end{bmatrix}, \tag{J.13b}$$

equation (J.2a) becomes

$$\begin{bmatrix} \hat{x}_{k+1|k} \\ \hat{\theta}_{k+1} \end{bmatrix} = \begin{bmatrix} A_k \hat{x}_{k|k-1} + B_k u_k \\ \hat{\theta}_k \end{bmatrix} + \begin{bmatrix} K_k \\ L_k \end{bmatrix} \left[z_k - C_k \hat{x}_{k|k-1} - D_k u_k \right], \tag{J.14}$$

which clearly corresponds to (2.117a) and (2.117b). By introducing

$$\begin{aligned} S_k &\triangleq R_k(\hat{\eta}_k, u_k) P_k R_k^T(\hat{\eta}_k, u_k) + \Sigma_e \\ &= \begin{bmatrix} C_k & H_k \end{bmatrix} \begin{bmatrix} P_{1_k} & P_{2_k} \\ P_{2_k}^T & P_{3_k} \end{bmatrix} \begin{bmatrix} C_k & H_k \end{bmatrix}^T + \Sigma_e \\ &= C_k P_{1_k} C_k^T + H_k P_{2_k}^T C_k^T + C_k P_{2_k} H_k^T + H_k P_{3_k} H_k^T + \Sigma_e, \end{aligned} \tag{J.15}$$

equation (J.2b) becomes

$$\begin{bmatrix} K_k \\ L_k \end{bmatrix} = \left[\begin{bmatrix} A_k & F_k \\ 0 & I \end{bmatrix} \begin{bmatrix} P_{1_k} & P_{2_k} \\ P_{2_k}^T & P_{3_k} \end{bmatrix} \begin{bmatrix} C_k & H_k \end{bmatrix}^T + \Sigma_{\bar{v}e} \right] S_k^{-1}, \tag{J.16}$$

which yields (2.117c) and (2.117e). Finally, P_{k+1} becomes in block form

$$\begin{aligned} \begin{bmatrix} P_{1_k} & P_{2_k} \\ P_{2_k}^T & P_{3_k} \end{bmatrix} &= \begin{bmatrix} A_k & F_k \\ 0 & I \end{bmatrix} \begin{bmatrix} P_{1_k} & P_{2_k} \\ P_{2_k}^T & P_{3_k} \end{bmatrix} \begin{bmatrix} A_k & F_k \\ 0 & I \end{bmatrix}^T + \begin{bmatrix} \Sigma_v & 0 \\ 0 & \Sigma_d \end{bmatrix} - \begin{bmatrix} K_k \\ L_k \end{bmatrix} S_k \begin{bmatrix} K_k \\ L_k \end{bmatrix}^T, \\ &= \begin{bmatrix} A_k P_{1_k} A_k^T + F_k P_{2_k}^T F_k^T + A_k P_{2_k} F_k^T + F_k P_{3_k} A_k^T & A_k P_{2_k} + F_k P_{3_k} \\ P_{2_k}^T A_k^T + P_{3_k} F_k^T & P_{3_k} \end{bmatrix} \\ &\quad + \begin{bmatrix} \Sigma_v & 0 \\ 0 & \Sigma_d \end{bmatrix} - \begin{bmatrix} K_k S_k K_k^T & K_k S_k L_k^T \\ L_k S_k K_k^T & L_k S_k L_k^T \end{bmatrix} \end{aligned} \tag{J.17}$$

from which (2.117f)-(2.117h) are obtained.

Appendix K

Assumptions

This appendix serves as a reference book for the various assumptions which have been used within this thesis and which are scattered throughout the chapters.

System assumptions:

AS1 The dynamic system is asymptotically stable, i.e. $A(q^{-1})$ has all zeros inside the unit circle.

AS2 All system modes are observable and controllable, i.e. $A(q^{-1})$ and $B(q^{-1})$ have no common factors.

AS3 The polynomial degrees n_a and n_b are known a priori with $n_b \leq n_a$.

AS4 The bilinearity $\mathcal{N}$ is chosen such that y_{0_k} is zero mean.

AS5 The polynomials $A(q^{-1})$ and $B(q^{-1})$ are of the same order, i.e. $n \triangleq n_a = n_b$, whilst the polynomials $C(q^{-1})$ and $D(q^{-1})$ are chosen such that $n_c = n_d - 1$.

Input assumptions:

AI1 The true input u_{0_k} is a zero-mean ergodic process and is persistently exciting of sufficiently high order.

AI2 The system input u_k is known exactly.

AI3 The true input u_{0_k} is a stationary zero-mean ergodic process with variance σ_{u_0}.

AI4 The system input u_k behaves in a manner, such that the bilinear system whose dynamics are characterised by the state transition matrix $A + Nu_k$ is stable.

AI5 The noise-free system input u_{0_k} has a rational spectrum, i.e. it can be described as an ARMA process of the form

$$D(q^{-1})u_{0_k} = C(q^{-1})f_k, \tag{K.1}$$

where f_k is a white noise zero mean random process and the polynomials $C(q^{-1})$ and $D(q^{-1})$ are defined, respectively, by

$$C(q^{-1}) \triangleq 1 + c_1 q^{-1} + \cdots + c_{n_c} q^{-n_c}, \tag{K.2a}$$
$$D(q^{-1}) \triangleq 1 + d_1 q^{-1} + \cdots + d_{n_d} q^{-n_d}. \tag{K.2b}$$

Noise assumptions:

AN1 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are zero-mean, ergodic, white noises with unknown variances, denoted $\sigma_{\tilde{u}}$ and $\sigma_{\tilde{y}}$, respectively, i.e.

$$\sigma_{\tilde{u}}\delta_{kl} \triangleq E\left[\tilde{u}_k \tilde{u}_l\right], \tag{K.3a}$$
$$\sigma_{\tilde{y}}\delta_{kl} \triangleq E\left[\tilde{y}_k \tilde{y}_l\right]. \tag{K.3b}$$

AN1a The sequence $\tilde{u}_k$ is a zero-mean, ergodic, white noise process with unknown variance $\sigma_{\tilde{u}}$.

AN1b The sequence $\tilde{y}_k$ is a zero-mean, ergodic noise process with unknown auto-covariance sequence $\{r_{\tilde{y}}(0), r_{\tilde{y}}(1), \cdots\}$.

AN2 The sequences $\tilde{u}_k$ and $\tilde{y}_k$ are mutually uncorrelated and also uncorrelated with both u_{0_k} and y_{0_k}.

AN3 The noise sequences v_k and e_k are zero mean, white, and satisfy

$$E\left[\begin{bmatrix} v_k \\ e_k \end{bmatrix}\begin{bmatrix} v_l^T & e_l^T \end{bmatrix}\right] = \begin{bmatrix} \Sigma_v^k & \Sigma_{ve}^k \\ \Sigma_{ve}^{k^T} & \Sigma_e^k \end{bmatrix}\delta_{kl}. \tag{K.4}$$

AN4 The initial state x_0 has the mean $\bar{x}_0$ with covariance matrix P_0. In addition, x_0 is independent of $\begin{bmatrix} v_k^T & e_k^T \end{bmatrix}^T$ for all k.

AN5 The quantities x_0, v_k and e_k are jointly Gaussian.

AN6a The noise sequences $\tilde{u}_k$, $\tilde{y}_k$ and w_k are assumed to be zero mean, white, independent of u_{0_k} and are characterised by the known covariance matrices

$$E\left[\begin{bmatrix} x_0 \\ \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix}\begin{bmatrix} x_0^T & \tilde{u}_l^T & \tilde{y}_l^T & w_l \end{bmatrix}\right] = \begin{bmatrix} P_0 & 0 & 0 & 0 \\ 0 & \Sigma_{\tilde{u}}^k & \Sigma_{\tilde{u}\tilde{y}}^k & 0 \\ 0 & \Sigma_{\tilde{u}\tilde{y}}^{k^T} & \Sigma_{\tilde{y}}^k & 0 \\ 0 & 0 & 0 & \Sigma_w^k \end{bmatrix}\delta_{kl}. \tag{K.5}$$

AN6b The noise sequences $\tilde{u}_k$, $\tilde{y}_k$ and w_k are assumed to be stationary, zero mean, white, independent of u_{0_k} and are characterised by the known covariance matrices

$$E\left[\begin{bmatrix} x_0 \\ \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix}\begin{bmatrix} x_0^T & \tilde{u}_l^T & \tilde{y}_l^T & w_l \end{bmatrix}\right] = \begin{bmatrix} P_0 & 0 & 0 & 0 \\ 0 & \Sigma_{\tilde{u}} & \Sigma_{\tilde{u}\tilde{y}} & 0 \\ 0 & \Sigma_{\tilde{u}\tilde{y}}^T & \Sigma_{\tilde{y}} & 0 \\ 0 & 0 & 0 & \Sigma_w \end{bmatrix}\delta_{kl}. \tag{K.6}$$

AN7 The noise sequences $\tilde{u}_k$, $\tilde{y}_k$ and w_k are assumed to be zero mean, white, independent of u_{0_k} and are characterised by the known covariance matrices

$$E\left[\begin{bmatrix} x_0 \\ \tilde{u}_k \\ \tilde{y}_k \\ w_k \end{bmatrix}\begin{bmatrix} x_0^T & \tilde{u}_l & \tilde{y}_l & w_l \end{bmatrix}\right] = \begin{bmatrix} P_0 & 0 & 0 & 0 \\ 0 & \sigma_{\tilde{u}} & \sigma_{\tilde{u}\tilde{y}} & 0 \\ 0 & \sigma_{\tilde{u}\tilde{y}} & \sigma_{\tilde{y}} & 0 \\ 0 & 0 & 0 & \sigma_w \end{bmatrix}\delta_{kl}. \tag{K.7}$$

Estimator assumptions:

AE1 The estimate of the input measurement noise variance $\hat{\sigma}_{\tilde{u}}^k$ 'varies slowly' with time.

AE2 The dimension of the instrument vector ζ_k is $n_a + n_b + 1$.

References

Anderson, B. D. O. & Moore, J. B. (1979), *Optimal Filtering*, Prentice-Hall, Englewood Cliffs, New Jersey.

Arulampalam, M. S., Maskella, S., Gordon, N. & Clapp, T. (2002), 'A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking', *IEEE Trans. on Signal Proc.* **50**(2), 174–188.

Beghelli, S., Guidorzi, R. P. & Soverini, U. (1990), 'The Frisch scheme in dynamic system identification', *Automatica* **26**(1), 171–176.

Björck, Å. (1996), *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia.

Björck, A. (1997), Newton and Rayleigh quotient methods for total least squares problems, *in* S. Van Huffel, ed., 'Recent advances in total least squares and errors-in-variables modeling', Siam, Philadelphia, USA, pp. 149–160.

CAPTAIN Toolbox for Matlab (2008), http://www.es.lancs.ac.uk/cres/captain/.

Carravetta, F., Germani, A. & Raimondi, M. (1997), 'Polynomial Filtering of Discrete-Time Stochastic Linear Systems with Multiplicative State Noise', *IEEE Trans. Autom. Contr.* **42**(8), 1106–1126.

Chang, P. S. & Wilson, A. N. (2000), 'Analysis of Conjugate Gradient Algorithms for Adaptive Filtering', *IEEE Trans. on. Signal Proc.* **48**(2), 409–418.

Chen, H. F. (2007), 'Recursive identification for multivariate errors-in-variables systems', *Automatica* **43**, 1234–1242.

Chen, H. F. & Yang, J. M. (2005), 'Strongly consistent coefficient estimate for errors-in-variables models', *Automatica* **41**, 1025–1033.

Chen, H., Sarkar, T. K., Dianat, S. A. & Brulé, J. D. (1986), 'Adaptive spectral estimation by the conjugate gradient method', *IEEE Trans. on Acoustics, Speech, and Signal Proc.* **34**(2), 272–284.

Comon, P. & Golub, G. H. (1990), 'Tracking a few extreme singular values and vectors in signal processing', *Proc. of IEEE* **78**(8), 1327–1343.

Davila, C. E. (1994), 'An efficient recursive total least squares algorithm for FIR adaptive filtering', *IEEE Trans. on Signal Proc.* **42**(2), 268–280.

De Koning, W. L. (1984), 'Optimal Estimation of Linear Discrete-time Systems with Stochastic Parameters', *Automatica* **20**(1), 113–115.

DeGroat, R. D., Dowling, E. M. & Linebarger, D. A. (1997), *The Digital Signal Processing Handbook*, CRC Press/IEEE Press, chapter Subspace Tracking.

Dennis, J. E. & Schnabel, R. B. (1996), *Numerical methods for unconstrained optimization and nonlinear equations*, Soc for Industrial & Applied Math.

Ding, F., Chen, T. & Qiu, L. (2006), 'Bias compensation based recursive least-squares identification algorithm for MISO systems', *IEEE Trans. on Circuits and Systems* **53**(5), 349–353.

Diversi, R., Guidorzi, R. & Soverini, U. (2003*a*), 'Algorithms for optimal errors-in-variables filtering', *Systems & Control Letters* **48**, 1–13.

Diversi, R., Guidorzi, R. & Soverini, U. (2003*b*), A new criterion in EIV identification and filtering applications, *in* 'Preprints 13th IFAC Symposium on System Identification', Rotterdam, The Netherlands, pp. 1993–1998.

Diversi, R., Guidorzi, R. & Soverini, U. (2005), 'Kalman filtering in extended noise environments', *IEEE Trans. Autom. Contr.* **50**, 1396–1402.

Diversi, R., Guidorzi, R. & Soverini, U. (2006), Yule-Walker equations in the Frisch scheme solution of errors-in-variables identification problems, *in* 'Proc. 17th Int. Symp. on Math. Theory of Networks and Systems', pp. 391–395.

Ekman, M. (2005), Modeling and Control of Bilinear Systems, PhD thesis, Uppsala University.

Favoreel, W., De Moor, B. & Van Overschee, P. (1999), 'Subspace Identification of Bilinear Systems Subject to White Inputs', *IEEE Trans. Autom. Contr.* **44**(6), 1157–1165.

Feng, D., Zhang, H., Zhang, X. & Bao, Z. (2001), 'An extended recursive least-squares algorithm', *Signal Processing* **81**(5), 1075–1081.

Feng, Y. T. & Owen, D. R. J. (1996), 'Conjugate gradient methods for solving the smallest eigenpair of large symmetric eigenvalue problems', *Int. J. for Numerical Methods in Engineering* **39**, 2209–2229.

Fnaiech, F. & Ljung, L. (1987), 'Recursive identification of bilinear systems', *Int. J. Control* **45**(2), 453–470.

Friedlander, B. (1984), 'The overdetermined recursive instrumental variable method', *IEEE Trans. on Autom. Contr.* **29**(4), 353–356.

Frisch, R. (1934), Statistical confluence analysis by means of complete regression systems, Technical Report 5, University of Oslo, Economics Institute, Oslo, Norway.

Geromel, J. C. (1999), 'Optimal linear filtering under parameter uncertainty', *IEEE Trans. on Signal Proc.* **47**(1), 168–175.

Gillijns, S. & De Moor, B. (2006), Linear Recursive Filtering with Noisy Input and Output Measurements, Technical Report ESAT-SISTA/TR 06-166, Katholieke Universiteit Leuven.

Golub, G. H. & Pereyra, V. (1973), 'The differentiation of pseudo-inverses and nonlinear least squares problems whose variables are separate', *Siam J. Numer. Anal.* **10**(2), 413–432.

Golub, G. H. & Pereyra, V. (2002), Separable Nonlinear Least Squares: the Variable Projection Method and its Applications, Technical Report SCCM-02-07, Stanford University, Stanford, USA.

Golub, G. H. & Van Loan, C. F. (1996), *Matrix Computations*, 3rd edn, Johns Hopkins University Press, Baltimore and London.

Guidorzi, R., Diversi, R. & Soverini, U. (2003), 'Optimal errors-in-variables filtering', *Automatica* **39**, 281–289.

Guidorzi, R. & Pierantoni, M. (1995), A new parametrization of Frisch scheme solutions, *in* 'Proc. of the 12th Int. Conf. on System Science', Wroclaw, Poland, pp. 114–120.

Hong, M. & Söderström, T. (2008), Relations between Bias-Eliminating Least Squares, the Frisch Scheme and Extended Compensated Least Squares Methods for Identifying Errors-in-Variables Systems, Technical Report 2008-008, Uppsala University, Uppsala, Sweden.

Hong, M., Söderström, T., Soverini, U. & Diversi, R. (2007), Comparison of three Frisch methods for errors-in-variables identification, Technical Report 2007-021, Uppsala University, Uppsala, Sweden.

Jazwinski, A. H. (1970), *Stochastic Process and Filtering Theory*, Academic Press, New York.

Kailath, T. & Sayed, A. H. (2000), *Linear Estimation*, 1st edn, Prentice Hall, Upper Saddle River, NJ, USA.

Kalman, R. E. (1960), 'A new approach to linear filtering and prediction problems', *Journal of Basic Engineering* **82D**(1), 35–45.

Kolmogorov, A. N. (1941), 'Interpolation and extrapolation', *Bull. de l'academie des sciences de U.S.S.R.* **Ser. Math. 5**, 3–14.

Kushner, H. J. & Yin, G. G. (2003), *Stochastic Approximation and Recursive Algorithms and Applications*, Applications of Mathematics, 2nd edn, Springer, New York.

Larkowski, T., Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2008), 'Novel algorithms based on conjunction of the Frisch scheme and extenden compensated least squares', in Proc. of UKACC Int. Control Conf., Manchester, UK.

Lawson, C. L. & Hanson, R. J. (1995), *Solving Least Squares Problems*, Classics in Applied Mathematics, 15 edn, SIAM.

Linden, J. G. & Burnham, K. J. (2008*a*), A recursive Frisch scheme algorithm for coloured output noise, *in* 'Proc. Int. Conf. on Informatics in Control, Automation and Robotics', Madeira, Portugal, pp. 163–170.

Linden, J. G. & Burnham, K. J. (2008*b*), Recursive Frisch scheme identification via variable projection, *in* 'Proc. CD-ROM 11th Mechatronics Forum Biennial International Conference', Limerick, Ireland.

Linden, J. G. & Burnham, K. J. (2008*c*), Some aspects on recursive Frisch scheme identification, *in* 'Computer Systems Engineering, Proc. 6th & 7th Polish British Workshop', pp. 176–187.

Linden, J. G., Larkowski, T. & Burnham, K. J. (2008), An improved recursive Frisch scheme identification algorithm, *in* 'Proc. 19th Int. Conf. on Systems Engineering', Las Vegas, USA, pp. 65–70.

Linden, J. G., Meyer, N., Vinsonneau, B. & Burnham, K. J. (2006), Recursive Frisch scheme identification incorporating adaptivity, *in* 'Proc. DVD-ROM 21st IAR & ACD Workshop', Nancy, France.

Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2007*a*), Errors-in-variables filtering approaches for bilinear systems, *in* A. Grzech, ed., 'Proc. of 16th Int. Conf. Systems Science', Vol. 1, Wroclaw, Poland, pp. 446–455.

Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2007*b*), Fast algorithms for recursive Frisch scheme system identification, *in* 'Proc. CD-ROM 22nd IAR & ACD Workshop', Grenoble, France.

Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2007*c*), An investigation of extended Kalman filtering in the errors-in-variables framework - A joint state and parameter estimation approach, *in* 'Proc. Int. Conf. on Informatics in Control, Automation and Robotics', Angers, France, pp. 47–53.

Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2008), Gradient-based approaches for recursive Frisch scheme identification, *in* 'Preprints of the 17th IFAC World Congress', Seoul, Korea, pp. 1390–1395.

Ljung, L. (1979), 'Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems', *IEEE Trans. on Automatic Control* **24**(1), 36–50.

Ljung, L. (1999), *System Identification - Theory for the user*, PTR Prentice Hall Information and System Sciences Series, 2nd edn, Prentice Hall, New Jersey.

Ljung, L. & Söderström, T. (1983), *Theory and Practice of Recursive Identification*, M.I.T. Press, Cambridge, MA.

Markovsky, I. & De Moor, B. (2005), 'Linear dynamic filtering with noisy input and output', *Automatica* **41**, 167–171.

MathWorks (2007), *Matlab documentation*, 2007b edn, The MathWorks.

Mathworks (2008), *System Identification Toolbox documentation*, The Mathworks.

Meyer, N., Linden, J. G., Vinsonneau, B. & Burnham, K. J. (2006), A recursive Frisch scheme approach for errors-in-variables system identification, *in* 'Proc. 18th Int. Conf. on Systems Engineering', Coventry, UK, pp. 281–286.

Mohler, R. R. (1991), *Nonlinear Systems - Volume 2: Applications to Bilinear Control*, Prentice Hall, New Jersey.

Pearson, R. K. (1999), *Discrete-Time Dynamic Models*, Oxford University Press, New York.

Pearson, R. K. & Kotta, U. (2004), 'Nonlinear discrete-time models: state-space vs. I/O representation', *J. of Process Control* **14**, 533–538.

Pintelon, R. & Schoukens, J. (2007), 'Frequency domain maximum likelihood estimation of linear dynamic errors-in-variables models', *Automatica* **43**(4), 621–630.

Polderman, J. W. & Willems, J. C. (1998), *Introduction to mathematical systems theory: a behavioral approach*, Texts in Applied Mathematics, Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Sagara, S. & Wada, K. (1977), 'On-line modified least-squares parameter estimation of linear discrete dynamic systems', *Int. J. Control* **25**(3), 329–343.

Sayed, A. & Kailath, T. (1994*a*), 'A state-space approach to adaptive RLS filtering', *IEEE Signal Processing Magazine* **11**(3), 18–60.

Sayed, A. & Kailath, T. (1994*b*), A state-space approach to adaptive RLS filtering (Expanded version), Technical report, University of California Los Angeles.

Söderström, T. (1981), 'Identification of stochastic linear systems in presence of input noise', *Automatica* **17**, 713–725.

Söderström, T. (2006), Extending the Frisch scheme for errors-in-variables identification to correlated output noise, Technical report, Uppsala University, Department of Information Technology, Uppsala, Sweden.

Söderström, T. (2007*a*), 'Accuracy analysis of the Frisch scheme for identifying errors-in-variables systems', *IEEE Trans. Autom. Contr.* **52**(6), 985–997.

Söderström, T. (2007*b*), 'Errors-in-variables methods in system identification', *Automatica* **43**(6), 939–958.

Söderström, T. (2008), 'Extending the Frisch scheme for errors-in-variables identification to correlated output noise', *Int. J. of Adaptive Control and Signal Proc.* **22**(1), 55–73.

Söderström, T., Hong, M. & Zheng, W. X. (2005), 'Convergence properties of bias-eliminating algorithms for errors-in-variables identification', *Int. J. of Adaptive Control and Signal Proc.* **19**, 703–722.

Söderström, T. & Mahata, K. (2002), 'On instrumental variable and total least squares approaches for identification of noisy systems', *Int. J. Control* **75**(6), 381–389.

Söderström, T., Soverini, U. & Mahata, K. (2002), 'Perspectives on errors-in-variables estimation for dynamic systems', *Signal Proc.* **82**(8), 1139–1154.

Söderström, T. & Stoica, P. (1989), *System identification*, Prentice Hall International, Hemel Hempstead, UK.

Sorenson, H. W. (1970), 'Least squares estimation: From Gauss to Kalman', *IEEE Spectrum* **7**, 63–68.

Stoica, P. & Söderström, T. (1982), 'Bias correction in least squares system identification', *Int. J. Control* **35**(3), 449–457.

Stoica, P. & Söderström, T. (1995), 'On the convergence properties of a time-varying recursion', *IEEE Signal Proc. Letters* **2**(5), 95–96.

Van Huffel, S., ed. (1997), *Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modelling*, SIAM, Philadelphia.

Van Huffel, S. & Lemmerling, P., eds (2002), *Total least squares and errors-in-variables modeling - Analysis, Algorithms and Applications*, Kluwer Academic Publishers.

Van Huffel, S. & Vandewalle, J. (1991), *The Total Least Squares Problem: Computational aspects and analysis*, SIAM.

Verdult, V. (2002), Nonlinear System Identification: A State-Space Approach, PhD thesis, University of Twente, Netherlands.

Wang, F. & Balakrishnan, V. (2002), 'Robust Kalman Filters for Linear Time-Varying Systems With Stochasatic Parametric Uncertainties', *IEEE Trans. on Signal Proc.* **50**(4), 803–813.

Wang, Z. & Qiao, H. (2002), 'Robust filtering for bilinear uncertain stochastic discrete-time systems', *IEEE Trans. on Signal Proc.* **50**(3), 560–567.

Wiener, N. (1964), *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*, The MIT Press.

Yang, H. (1993), 'Conjugate gradient methods for the Rayleigh quotient minimization of generalized eigenvalue problems', *Computing* **51**, 79–94.

Yaz, E. (1992), 'Robust, exponentially fast state estimator for some non-linear stochastic systems', *Int. J. Systems Sci.* **23**(4), 557–567.

Young, P. C. (1974), 'Recursive approaches to time series analysis', *Bull. of Inst. Maths and its Applications* **10**, 209–224.

Young, P. C. (1984), *Recursive Estimation and Time Series Analysis*, Springer-Verlag, Berlin.

Young, P. C. (1999), 'Nonstationary time series analysis and forecasting', *Progress in Environmental Science* **1**, 3–48.

Young, P. C. (2000), *Nonlinear and Nonstationary Signal Processing*, Cambridge University Press, Cambridge, UK, chapter Stochastic, dynamic modelling and signal processing: time variable and state dependent parameter estimation, pp. 74–114.

Young, P. C. (2002), *Encyclopedia of Life Support Systems*, Vol. 6.43: Control Systems, Robotics and Automation, Oxford, Oxford, UK, chapter Identification of time varying systems.

Young, P. C., McKenna, P. & Bruun, J. (2001), 'Identification of non-linear stochastic systems by state dependent parameter estimation', *Int. J. Control* **74**(18), 1837–1857.

Young, P. C., Taylor, C. J. & Chotai, A. (2009), *True Digital Control*, Taylor-Francis, London, chapter 8.

Zheng, W. X. (1998), 'Transfer function estimation from noisy input and output data', *Int. J. of Adaptive Control and Signal Proc.* **12**, 365–380.

Zheng, W. X. (1999), 'On least-squares identification of stochastic linear systems with noisy input-output data', *Int. J. Adaptive Control and Signal Proc.* **13**, 131–143.

Zheng, W. X. (2000), 'Parametric identification of linear noisy input-output systems', *Cybernetics and Systems* **31**, 803–816.

Zheng, W. X. & Feng, C. B. (1989), 'Unbiased parameter estimation of linear systems in the presence of input and output noise', *Int. J. of Adaptive Control and Signal Proc.* **3**, 231–251.