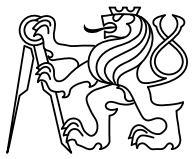
Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Radioelectronics



Low Complex PVT Estimation Using Factor Graphs

Doctoral Thesis

Ing. Ondřej Jakubov

June 2013, Prague

Supervisor:	Doc. Dr. Ing. Pavel Kovář
Supervisor-Specialist:	Ing. Petr Kačmařík, Ph.D.
Ph.D. Programme:	Electrical Engineering and Information Technology
Branch of Study:	2601V010 Radioelectronics

Annotation

The thesis first summarizes various position-velocity-time (PVT) estimation and filtering algorithms in global navigation satellite systems (GNSS), second proposes a novel low complex PVT filtering algorithm using the factor graph (FG) theory, and third delivers a simulation and experimental comparison of the new method with the existing ones. The overviewed algorithms are either implemented in commercial receivers or a subject to research due to their computational complexity and insufficient verification of their precision and robustness. These classical estimation algorithms include least squares (LS) and weighted least squares (WLS), whereas the Bayesian filtering algorithms include extended Kalman filter (EKF), unscented Kalman filter (UKF), grid-based filter (GF), and particle filter (PF).

The operations of the PVT estimation/filtering algorithms used in practice involve matrix operations which complexity grows significantly with the increasing number of measurements. On the other hand, the algorithms such as UKF, GF, PF do not suffer from such dependency issue even though they work on vector data, but their performance is strictly violated if a relatively large number of representative samples are not selected.

Here, we propose a novel suboptimal iterative method that operates only on scalars, employs simple arithmetic operations and is fully distributed. Hence, the algorithm can be implemented in parallel processing units, each for a tracked satellite. The scalar nature of the algorithm and simplicity of the arithmetic operations facilitate the requirements on the navigation processor which does not need to include libraries for matrix operations any more. The distributed nature additionally enables to move the algorithm to hardware logic and hence minimize the CPU's load.

Simulations comparing precision and convergence of the proposed FG-based PVT filtering method with the most commonly implemented method - EKF are delivered in the text. It is demonstrated that the FG-based method features similar precision and stability at number of visible satellites larger than 16. These methods have already been implemented into a real-time GNSS receiver developed at CTU and tested. A case study is presented.

Nomenclature

ADC	Analog-to-Digital Convertor, page 25
ALU	Arithmetic and Logic Unit, page 40
AOA	Angle-of-Arrival, page 11
BER	Bit Error Ratio, page 30
BOC	Binary-Offset Keying, page 23
BPSK	Binary-Phase-Shift Keying, page 23
CDMA	Code Division Multiple Access, page 23
CPU	Central Processing Unit, page 11
CRC	Cyclic Redundancy Check, page 29
CRLB	Cramer-Rao Lower Bound, page 14
CS	Coordinate System, page 19
DLL	Delay-Locked Loop, page 27
DPE	Direct Poisition Estimation, page 38
DS SS	Direct-Sequence Spread-Spectrum, page 23
DSP	Digital Signal Processor, page 24
E/L	Eearly/Late, page 28
ECEF	Earth-Centred Earth-Fixed, page 19
EGNOS	European Geostationary Navigation Overlay Service, page 9

EKF	Extended Kalman Filter, page 10
EM	Expectation-Maximation, page 60
ENU	East North Up, page 106
FBS	FBS, page 27
FDMA	Frequency Division Multiple Access, page 21
FFT	Fast Fourier Transform, page 27
FG	Factor Graph, page 10
FLL	Frequency-Locked Loop, page 27
FPGA	Field Programmable Gate Array, page 11
GF	Grid-based Filter, page 10
GLONASS	GLObalnaja NAvigacionnaja Sputnikovaja Sistema, page 9
GNSS	Global Navigation Satellite System, page 9
GPS	Global Positioning System, page 9
GPU	Graphic Processing Unit, page 11
H-GNSS	Hybrid GNSS (often combined with cellur mobile networks), page 9
IF	Intermediate Frequency, page 20
IID	Independent and Identically Distributed, page 61
LD	Low Dynamics, page 117
LDPC	Low-Density-Parity-Check, page 27
LMDM	Linear Multi-Dimensional Modulation, page 13
LMMSE	Linear Minimum Mean Square Error, page 70
LNA	Low Noise Amplifier, page 25
LO	Local Oscillator, page 21

LS	Least Squares, page 10
LSE	Least Squares Estimator, page 62
MD	Moderate Dynamics, page 117
MEDLL	Multipath Estimating Delay Locked Loop, page 25
ML	Maximum Likelihood, page 11
MMSE	Minimum Mean Square Error, page 19
MPA	Message Passing Algorithm, page 11
MS	Mobile Station, page 11
MSE	Mean Square Error, page 53
MVU	Minimum Variance Unbiased, page 56
NLOS	Non-Line-of-Sight Positioning, page 11
PDF	Probability Density Function, page 10
PDOP	Position Dilution of Precision, page 31
PF	Particle Filter, page 10
PLL	Phase-Locked Loop, page 27
PSD	Power Spectral Density, page 22
PVT	Position, Velocity, and Time, page 9
RB	Radio Beacon, page 10
RNSS	Regional Navigation Satellite Systems, page 9
SBAS	Satellite-Based Augmentation System, page 9
SDR	Software-Defined Radio, page 11
SoL	Safety-of-Live, page 32
SPA	Sum-Product Algorithm, page 10

STVP	Slowly Time-Varying Parameters, page 23
SV	Space Vehicle, page 10
SVD	Singular Value Decomposition, page 97
TDOA	Time-Difference-of-Arrival, page 11
TOA	Time-of-Arrival, page 9
TPE	Total Position Error, page 115
TTFF	Time To First Fix, page 25
TVE	Total Velocity Error, page 115
UCorIP	Universal Correlator IP, page 45
UKF	Unscented Kalman Filter, page 10
UT	Unscented Transform, page 78
UWB	Ultra-Wide Band, page 11
WAAS	Wide Area Augmentation System, page 9
WGN	White Gaussian Noise, page 22
WLS	Weighed Least Squares, page 10
WNav	The Witch Navigator, page 11
WSS	Wide-Sense Stationary, page 61

Contents

Annotation **1**

Nomenclature **2**

Introduction **9**

1 GNSS SW Receiver and Architectures **18**
1.1 Signal Processing Algorithms in GNSS Receiver 19
1.1.1 Position Estimation 19
1.1.2 Velocity Estimation 20
1.1.3 Channel Model 22
1.1.4 Signal Structure 23
1.1.5 Signal Conditioning 24
1.1.6 Modularization of Traditional GNSS Receiver 26
1.1.6.1 Signal Acquisition 26
1.1.6.2 Signal Tracking and Bit Synchronization 27
1.1.6.3 Frame Synchronization, Data Demodulation and Decoding 29
1.1.6.4 Navigation Data Handling and Pseudorange Formation 30
1.1.6.5 PVT Estimation/Filtering 31
1.2 Receiver Architectures 32
1.2.1 Conventional Architecture 33
1.2.1.1 Measurement Equation for Estimation of Pseudoranges and Pseudorange Rates 33
1.2.1.2 Measurement Equation for PVT Estimation 35
1.2.1.3 Estimator Decomposition 36
1.2.2 Vector Tracking Architecture 36
1.2.3 Direct Positioning Architecture 37
1.3 Receiver Hardware Concepts 38

1.3.1 Traditional Concept with Real-Time Hardware Correlators . . 38
1.3.2 Software-Defined-Radio Concept 39

2 Testing Platform - WNav **42**
2.1 The Witch Navigator Project 42
2.2 Open Source Philosophy 43
2.3 Hardware 43
2.4 Software 47
2.5 Current State 49
2.6 Future Plans 49
2.7 Student's Project Tasks and Responsibilities 51

3 Overview of Estimation Theory **53**
3.1 Problem Definition 53
3.2 Classical Estimators 56
3.2.1 Minimum Variance Unbiased (MVU) Estimator and Cramer-Rao Lower Bound (CRLB) 56
3.2.2 Best Linear Unbiased Estimator (BLUE) 59
3.2.3 Maximum Likelihood Estimator (MLE) 60
3.2.4 Least Squares (LS) 62
3.2.5 Method of Moments 66
3.3 Bayesian Estimators - Minimum Mean Square Error (MMSE) Estimator and Maximum a Posteriori (MAP) Estimator 66
3.3.1 Linear MMSE Estimator and Wiener Filter 70
3.3.2 Kalman Filter (KF) 71
3.3.3 Extended Kalman Filter (EKF) 73
3.4 Posterior Cramer-Rao Lower Bound (PCRLB) 74

4 Nonlinear Bayesian Filters **77**
4.1 Unscented Kalman Filter (UKF) 78
4.1.1 Unscented Transform (UT) 78
4.1.2 Unscented Kalman Filter (UKF) 79
4.2 Grid-Based Methods 80
4.3 Particle Filtering 81
4.3.1 Monte Carlo Integration 82
4.3.2 Importance Sampling 83
4.3.3 Sequential Importance Sampling 83
4.3.4 Resampling 85
4.3.5 Particle Filter with Bootstrap Filter 85

5 Theory of FGs and the SPA **87**

6 Bayesian Filtering on the FG **91**
6.1 Kalman Filter on the FG . . . 92
6.2 Extended Kalman Filter on the FG . . . 96
6.3 Proposed Scalar Iterative Kalman Filter on the FG . . . 96
6.4 Proposed Scalar Iterative Extended Kalman Filter on the FG . . . 101

7 Overview of PVT Algorithms **103**
7.1 Classical Estimators . . . 104
7.1.1 Cramer-Rao Lower Bound . . . 104
7.1.2 Least Squares . . . 104
7.1.3 Weighted Least Squares . . . 105
7.1.4 Proposed Scalar Iterative Weighted Least Squares . . . 106
7.2 Bayesian Estimators . . . 106
7.2.1 Motion Models . . . 106
7.2.2 Posterior Cramer-Rao Lower Bound . . . 108
7.2.3 Extended Kalman Filter . . . 109
7.2.4 Scalar Iterative Extended Kalman Filter . . . 109
7.2.5 Unscented Kalman Filter . . . 110
7.2.6 Particle Filter (Bootstrap Filter) . . . 112

8 Simulation and Experimental Results **113**
8.1 Simulation - Scalar Tracking Architecture . . . 114
8.1.1 Simulation Scenarios . . . 114
8.1.2 Convergence . . . 115
8.1.3 Accuracy . . . 116
8.1.4 Summary . . . 116
8.2 Simulation - Vector Tracking Architecture . . . 117
8.2.1 Simulation Methodology . . . 117
8.2.2 Results . . . 121
8.2.3 Implementation Aspects . . . 121
8.3 Experiment - Scalar Tracking Architecture . . . 122
8.3.1 Methodology . . . 122
8.3.2 Results . . . 122

Conclusion **128**

A Complexity of Matrix and Vector Operations **141**

Introduction

Global Navigation Satellite Systems (GNSS) The first *global navigation satellite systems* (GNSS) - US GPS [1–3], Russian GLONASS [4] were originally developed for military applications. However, their civil liberalization opened new prospective of these systems to be employed in civil aviation, maritime, geodesy, car navigation and others. High performance, low size and power consumption of today's electronic circuits enabled incorporation of GNSS receivers in hand-held devices.

The widespread dependency on GNSS systems, massive market penetration, increasing demand on position awareness with higher precision, availability, continuity and integrity motivated governments all over the world to modernize their positioning systems (GPS, GLONASS) or spawn new ones - European Galileo [5], Chinese BeiDou [6]. In parallel, *satellite-based augmentation systems* (SBAS) and *regional navigation satellite systems* (RNSS) were developed to support areas of countries or parts of continents - US WAAS [7, 8], European EGNOS [9, 10], Japanese QZZS [11] etc. Larger number of visible space vehicles, monitoring of local atmospheric effects and GNSS system performance, possibility of higher data rate transmission, and combination of these systems all contribute to much higher precision and robustness of receiver *position, velocity, and time* (PVT) estimation.

The *time-of-arrival* (TOA) measurements, which GNSS positioning is designed for [12, 13], are negatively affected by unknown biases caused by the earth's atmosphere, inaccurate satellite ephemeris and clock prediction. Nonetheless, these biases are common to receivers close to each other. If one of the receivers knows precisely its own position, it can estimate the sum of the biases to each visible satellite and augment other receivers with this information which is named as *differential GNSS*. Some areas are equipped with local fixed transmitters with identical signal structure as the GNSS have to augment the user with other radio beacons fostering geometric diversity with high power signals that can penetrate indoor buildings. Such transmitters are called *pseudolites*. Taking benefit of cellular mobile network TOA measurements, *hybrid GNSS* (H-GNSS) positioning is implemented by several chip vendors.

Measurement Data Increase and Nonlinear Effect The constantly increasing number of visible space vehicles (SV) and other radio beacons (RB), such as pseudolites and cellular mobile stations, challenges not only the design of digital signal processors providing standard outputs of *pseudoranges*, *pseudorange rates* and *navigation data* [14–19], but also the design of the position-velocity-time fusion algorithm handling large vector data, various coordinate systems and time references. The operations of the PVT estimation/filtering algorithms used in practice involve matrix operations which complexity grows significantly with the increasing number of measurements [20, 21]. These algorithms, including *least squares* (LS), *weighted least squares* (WLS), *extended Kalman Filter* (EKF), strictly rely on first order Taylor linearization of the geometry matrix composed of position vectors between the user and the radio beacons. The linearization works well for distant SVs and low user dynamics. If distances to narrow RBs are measured or the user maneuvers quickly, the geometry changes rapidly with respect to the time step of PVT estimation and the basic assumptions of the model simplification are violated.

The algorithms such as *unscented Kalman Filter* (UKF), *grid-based filter* (GF), and *particle filter* (PF) can model the nonlinearity based on the representation of the probability density function (PDF) by a finite number of samples [22–27]. These methods are mostly of quadratic or linear dependency on the number of visible RBs even though they work on vector data, but their performance is strictly corrupted if a relatively large number of representative samples are not selected.

Factor Graph Modeling In this study, we introduce a novel PVT estimation and filtering method derived from a graphical representation of the relations among the system variables using the *factor graph* (FG) framework [28, 29]. We first model the relations among the vector variables and derive the well known matrix algorithms such as WLS and EKF. We then split the vector nodes into scalar nodes, thus creating cycles in the graph, and derive novel distributed iterative algorithms with scalars. We then investigate precision, convergence, and complexity of these algorithms. In the study, we limit ourselves to Gaussian PDF representation (=first order Taylor representation), measurements only to distant SVs and user velocity dynamic model.

Factor graph modeling and so called *sum-product algorithm* (SPA) were chosen since a wide variety of algorithms developed in artificial intelligence, signal processing, and digital communications can be derived as specific instances of the SPA, operating in an appropriately chosen factor graph [28, 29]. Examples include iterative decoding of low-density-parity-check codes (LDPC) and turbo codes, the Viterbi algorithm, the Kalman filter, and certain fast Fourier transform (FFT) algorithms.

Factor graphs were first adopted to localization of a mobile station (MS) in wireless communication systems in 2003 [30, 31]. The MS therein estimates signal TOA from base stations with standard radius of 5 km. After a simple initialization, 2-D position is iteratively estimated using a simple easy-to-implement message-passing algorithm (MPA) with relatively high precision in comparison with the complex maximum likelihood (ML) algorithm. Later contributions incorporate similar approaches to time-difference-of-arrival (TDOA) or angle-of-arrival (AOA) localization [32], [33]. In [34], the authors propose a method for non-line-of-sight (NLOS) positioning in wireless communications using FG. The messages are therein represented by samples of the estimated PDF. The update rules for the vertices are accomplished using importance sampling. A universal algorithm based on FG, called SPAWN, has been developed for cooperative localization in wireless networks [35]. This promising algorithm takes into account the history of the measurements, motion model and is fully distributed. The tests were conducted using ultra-wideband (UWB) radios indoor buildings. A hybrid GNSS/UWB extension for cooperative localization is presented in [36]. However, this algorithm operates on vectors.

The Witch Navigator As a parallel line of the PhD study topic, the student contributes to an SDR receiver project, named *the Witch Navigator* (WNav) [37–41]. The aim of the project is to develop a software GNSS receiver that can process most of the civil GNSS signals. Current version of the receiver software can process GPS L1 and GLONASS L1&L2 signals. The student is responsible for data demodulation&decoding, pseudorange formation, navigation data storage and PVT estimation. The receiver is perfectly matched to the thesis requirements of algorithm's verification under real scenarios, therefore served as the main testing platform for the conducted research. Its hardware concept further allows real-time implementation of advanced receiver architectures such as vector tracking, direct positioning using either FPGA *universal correlators* [42], powerful *multi-core CPUs* or massively parallelized *graphic processing units* (GPUs), which are gradually being integrated with CPUs and deployed into low power hand-held devices.

Use Case	Benefits	How
CPU's offload to HW logic	Low-end CPU	HDL directly, C-to-HDL, C-to-RTL
	SW/HW tradeoffs	
	Higher number of SVs to track	
	Higher stability at high dynamics	Faster PVT updates in a vector tracking architecture
	Avoiding inertial sensors	
CPU's implementation	Smaller code size	No matrix library in use
	Smaller data memory	
	Lower OS requirements	

Table 1: Use cases and benefits of the proposed algorithm

Use Cases of the Proposed Algorithm

The primary motivation of the thesis was to find an algorithm which can facilitate the CPU's requirements on the PVT algorithm by any means. The found algorithm enables us to offload the CPU to HW logic thanks to its distributive nature or if implemented in the CPU, no matrix operations are needed.

The former enables us to use a lower-end CPU and tradeoffs between the SW/HW can be made using either the new algorithm or the existing algorithm. Dedicated hardware logic may incorporate more satellite channels than a low-end CPU. The longer development of programmable logic is about to be overcome in FPGAs with new tools by Xilinx (C-to-HDL) and Altera (C-to-RTL) who claim to instantiate HDL entities directly from C code using special directives. The bus communication, address map a device drivers will be autogenerated.

In higher dynamic applications, the PVT estimates can be generated more frequently if they run in HW compared to the sequential processor. This becomes beneficial in vector tracking architecture where the PVT estimates are used to control the tracking loops, hence increasing the system stability and avoiding need for inertial sensors.

If the algorithm runs under a CPU, no matrix libraries imply smaller code size and lower requirements on the operating system or standalone application. Thanks to the iterative nature of the proposed algorithm and the necessity to store only means and variances, the data memory requirements are lowered.

Use cases and benefits of the proposed algorithm are summarized in Table 1.

Contribution of the PhD Thesis

The contributions of the PhD study can summarized as follows:

- Summary of various stand-alone GNSS PVT estimation/filtering methods is given, including those being implemented in commercial receivers and those being a subject to research.
- The FG-based method of 2-D localization of MS in mobile communications [30, 31] is extended to account for GNSS SVs and other radio beacons, receiver clock offsets to the GNSS systems and velocity estimation.
- Novel FG-based Bayesian filtering method for various PDF representations, is proposed. The method is distributed, therefore suitable for implementation in parallel processing units.
- FG-based Bayesian filtering method with Gaussian PDF is further developed and analyzed. It employs no matrix operations, and uses simple arithmetic operations such as summation, subtraction, multiplication and division. Compromises can be made about complexity, precision, and robustness.
- Simulation and measurement results of this FG-based algorithm for Gaussian PDFs and GNSS SVs are delivered. Complexity, precision, convergence comparison with the EKF and GPS L1 SVs are available.
- Significant contribution to the GNSS SDR receiver project WNav by the student enables other researches to verify their new algorithms, which were tested only in a simulation, in real scenarios and fill thus a gap between theory and engineering practice.
- The bit-true Monte Carlo simulations carried out at the sampling rate to investigate behavior of tracking loops and other signal processing modules involving correlation can be bypassed by so called *semi-analytic models*. The student has contributed to this topic by establishing a universal framework for description of GNSS modulations by *linear multi-dimensional modulations* (LMDM), extended the simulation methods to account for feed-back delay, phase noise, finite bandwidth, and slowly fading multipath effects.

Document Structure

In Chapter 1, we first very generally address the topic of position-velocity-time (PVT) estimation in satellite navigation, defining basic quantities and establishing notation necessary throughout the text. We will show how the process of PVT estimation is modularized in a conventional receiver, discussing complexity of each step and specifying the study topic. Not always hold the basic assumptions about the communication channel and assumptions about the physical conditions. On that account, we list negative phenomena that a GNSS receiver is subjected to, sum up performance properties as well as properties important for future receiver development.

In Chapter 2, the testing platform for the study is introduced. Starting with the Witch Navigator (WNav) project description, open source philosophy, receiver hardware and software, we overview the current state of the project and discuss how it is used within this work. To conclude the topic of receivers, we forecast future WNav features and pinpoint student's project tasks and responsibilities. The project is described in [16].

Chapter 3 gives an overview of basic estimation theory concepts and properties. The chapter is a brief digest of [43] with minor modifications. In Chapter 4, we briefly discuss the sample-based nonlinear Bayesian filtering algorithms - UKF, GB filter, PF. The theory of factor graphs (FG) and the sum-product algorithm (SPA), the framework for derivation of the proposed algorithms, are summarized in Chapter 5. An example FG with SPA is added to facilitate the problem understanding. This chapter has been inspired by [28, 29]. Chapter 6 shows how the Kalman filter, or the extended Kalman filter can be derived using FG. The approach to the proposed suboptimal scalar iterative KF or EKF is then presented.

In Chapter 7, we overview the existing PVT estimation/filtering methods - namely LS, WLS, EKF, UKF, and PF. Cramer-Rao lower bound (CRLB) for the classical estimator and posterior CRLB for Bayesian estimator are derived. The adopted user motion models, based on Gauss-Markov process modeling, are also discussed. In the same chapter, we derive the proposed FG-based method of PVT estimation/filtering with Gaussian PDF representation. Simulation and experimental results are presented for this case in comparison with the EKF in Chapter 8.

The hierarchy of the document, purpose of the chapters, novelty and author's publications are summarized in Table 2.

Chapter	Purpose	Novelty	Publications
1 GNSS Receiver	Define quantities	GNSS Modulation	[44, 45]
	Define system model		
	Sum up architectures		
2 HW Testing Platform	Justify the Experiments	40% of SW	[16, 38–40, 46]
3 Estimation Theory	Theory of "commercial" WLS, EKF + (P)CRLB		
4 Nonlinear Bayesian Filters	Theory of "scientific" UKF, GF, PF		
5 Theory of FG and SPA	Background to derive FG-based filters		
6 Bayesian Filtering on FG	Derive existing (E)KF on FG	N-dimensional extension of scalar update rules for Gaussian densities to [49]	[47, 48]
	Derive novel scalar iterative (E)KF	New algorithm with no matrix operations, simple arithmetic, is distributed ⇒ can run in HW	[47, 48]
7 Overview of PVT Algorithms	Substitute model from Chap. 1 to methods from Chap. 3, 4, 6	Final formulae of the new algorithm for GNSS PVT filter	[48]
8 Simulations and Experiments	Simulate algorithms from Chap. 7	Convergence, accuracy, scenario analysis	[48]
	Perform a case study with HW platform from Chap. 2	Tests in a real-time GNSS receiver	[48]

Table 2: Hierarchy of the document, purpose of the chapters, novelty and author's publications

Notation

We denote vectors and matrices with bold emphasize throughout the text. All vectors are column vectors. We denote ith element of vector $\mathbf{a}$ as $[\mathbf{a}]_i$. Similarly, we denote the element at ith row and jth column of matrix $\mathbf{A}$ as $[\mathbf{A}]_{i,j}$. To denote ith row of matrix $\mathbf{A}$, we use $[\mathbf{A}]_{i,:}$, for jth column $[\mathbf{A}]_{:,j}$. All estimates of values are denoted with hat - an estimate of vector $\mathbf{a}$ is denoted as $\hat{\mathbf{a}}$. The mean value of random vector $\mathbf{a}$ is denoted as $\mathrm{E}[\mathbf{a}] = \boldsymbol{\mu}_\mathbf{a}$, the covariance matrix as $\mathbf{C}_\mathbf{a} = \mathrm{E}\left[(\mathbf{a} - \boldsymbol{\mu}_\mathbf{a})(\mathbf{a} - \boldsymbol{\mu}_\mathbf{a})^T\right]$ or equivalently $\mathbf{C}_{\mathbf{aa}}$. Additionally, the covariance matrix between two different vectors of the same size, say $\mathbf{a}$ and $\mathbf{b}$, is denoted as $\mathbf{C}_{\mathbf{ab}} = \mathrm{E}\left[(\mathbf{a} - \boldsymbol{\mu}_\mathbf{a})(\mathbf{b} - \boldsymbol{\mu}_\mathbf{b})^T\right]$. Diagonal matrices with elements $a_{1,1}, \ldots, a_{N,N}$, $N \in \mathbb{N}$ on the main diagonal are denoted as

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & 0 & \ldots & 0 & 0 \\ 0 & a_{2,2} & \ddots & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & & \ddots & a_{N-1,N-1} & 0 \\ 0 & 0 & \ldots & 0 & a_{N,N} \end{bmatrix} \triangleq \mathrm{diag}(a_{1,1}, \ldots, a_{N,N}). \tag{1}$$

If vector $\mathbf{a}$ represents a measurement in additive noise $\mathbf{w}$, then asterisk denotes the noiseless value $\mathbf{a}^*$

$$\mathbf{a}^* = \mathbf{a} - \mathbf{w}. \tag{2}$$

With notation $\mathcal{N}_\mathbf{a}(\boldsymbol{\mu}_\mathbf{a}, \mathbf{C}_\mathbf{a})$ we mean that random vector $\mathbf{a}$ has multivariate Gaussian (normal) distribution with mean $\boldsymbol{\mu}_\mathbf{a}$ and covariance matrix $\mathbf{C}_\mathbf{a}$. The inequality between two matrices means elementwise inequalities, e.g.

$$\mathbf{A} > \mathbf{B} \tag{3}$$

means $\forall i \in \{1, \ldots M\} \wedge \forall j \in \{1, \ldots N\}$ it holds that

$$[\mathbf{A}]_{i,j} > [\mathbf{B}]_{i,j} \tag{4}$$

if $\mathbf{A}$, $\mathbf{B}$ are $M \times N$ matrices. Time average of either continuous time $a(t)$ variable or discrete variable a_n is denoted with $\mathrm{Av}[.]$ such that

$$\mathrm{Av}[a(t)] = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} a(t)\mathrm{d}t \tag{5}$$

$$\mathrm{Av}[a_n] = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} a_n. \tag{6}$$

The operator $\mathrm{AvE}\left[.\right]$ is then a composition of ensemble and average $\mathrm{AvE}\left[.\right] = \mathrm{E}\left[\mathrm{Av}\left[.\right]\right] = \mathrm{Av}\left[\mathrm{E}\left[.\right]\right]$. The first partial derivative of M-dimensional vector function $\mathbf{g}\left(\boldsymbol{\theta}\right) = \left[g_1 \dots g_M\right]$ with respect to p-dimensional vector $\boldsymbol{\theta} = \left[\theta_1 \dots \theta_p\right]$, Jacobian matrix, is denoted as follows

$$\nabla_{\boldsymbol{\theta}} \mathbf{g}\left(\boldsymbol{\theta}\right) = \frac{\partial \mathbf{g}\left(\boldsymbol{\theta}\right)}{\partial \boldsymbol{\theta}} \tag{7}$$

$$= \begin{bmatrix} \frac{\partial g_1}{\partial \theta_1} & \cdots & \frac{\partial g_1}{\partial \theta_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_N}{\partial \theta_1} & \cdots & \frac{\partial g_N}{\partial \theta_p} \end{bmatrix}. \tag{8}$$

Partitioning vector $\boldsymbol{\theta}$ for $1 < r < p$ as

$$\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta}_1 \\ \boldsymbol{\theta}_2 \end{bmatrix} = \begin{bmatrix} r \times 1 \\ (p-r) \times 1 \end{bmatrix} \tag{9}$$

we define the second order partial derivative of $\mathbf{g}\left(\boldsymbol{\theta}_1,\, \boldsymbol{\theta}_2\right)$ with respect to $\boldsymbol{\theta}_2$ and to $\boldsymbol{\theta}_1$, respectively, as

$$\nabla^{\boldsymbol{\theta}_2}_{\boldsymbol{\theta}_1} \mathbf{g}\left(\boldsymbol{\theta}_1,\, \boldsymbol{\theta}_2\right) = \nabla_{\boldsymbol{\theta}_1} \left[\nabla^T_{\boldsymbol{\theta}_2} \mathbf{g}\left(\boldsymbol{\theta}_1,\, \boldsymbol{\theta}_2\right)\right]. \tag{10}$$

If $g\left(x_1, \dots x_K\right)$ is a scalar function of variables $x_1, \dots, x_K$, we denote the integral over all the variables except for x_k where $1 \leq k \leq K$ as

$$\int_{\sim x_k} g\left(x_1, \dots x_K\right) \mathrm{d}x_1 \dots \mathrm{d}x_K = \int \dots \int g\left(x_1, \dots x_K\right) \underbrace{\mathrm{d}x_1 \dots \mathrm{d}x_K}_{\text{except for } x_k}. \tag{11}$$

In the factor graph theory, we denote a message from variable node v to factor node F as $\lambda_{v \to F}$, and message from factor node F to variable node v as $\mu_{F \to v}$.

Chapter 1

GNSS Software Receiver and its Architectures

In this a very general chapter, we discuss various aspects of a design of a GNSS receiver. We deliver an approach describing all the signal processing steps in a GNSS receiver assuming reader's background in digital communications [50], spread spectrum communications [51], estimation and detection theory [43, 52], and the theory of synchronization and equalization [53, 54]. It crucial to understand the overall process of the signal processing up to the PVT estimation/filtering to efficiently design algorithms. A limited overview might result in a situation where effort would be devoted to improve performance of an algorithm in a specific signal processing step that could be much easily improved by a simple improvement in another step. Such would be the case when discussing various receiver architectures in Section 1.2. Tracking sensitivity of a GPS L1 C/A receiver can be improved by 7 dB using the vector tracking architecture with prefilters [55] that introduces no significant resource overhead, but its implementation requires good knowledge of the signal tracking and PVT estimation algorithms and all aspects connected with it. In Section 1.3.2, we discuss common receiver hardware concepts. The tendency of integration highly parallel DSPs with powerful CPUs on a chip opens new possibilities of implementation so called direct positioning methods that need to operate at signal samples iteratively which is not possible with traditional GNSS receiver concepts. The direct positioning methods are very advanced - all the signal processing steps merge. The engineers are challenged to understand the overall system well.

1.1 Signal Processing Algorithms in GNSS Receiver

In this section, we firstly address the problem of position and velocity estimation in GNSS. We do not derive methods of estimation based on particular criteria such as least squares, maximum likelihood, minimum mean square error (MMSE) etc. Instead, we state the measurement equations that are used for such estimations. The specific algorithms will be discussed in Chapter 7. Then, we introduce the considered channel model and the signal structure to continue with discussion on signal conditioning and description of modularization of a traditional receiver. It is unavoidable to mention the aspects of all GNSS signal processing steps of a traditional receiver, since they may influence the final PVT estimation in different ways. Concerning the channel model, we do not assume phenomena such as multipath, phase noise, fadings, but we discuss how these are dealt with a traditional receiver at a very top level.

1.1.1 Position Estimation

Let $\mathbf{x}_{S,i} = [x_{S,i}\, y_{S,i}\, z_{S,i}]^T$ denote the position of ith SV at the time of transmission $t_{S,i}$ according to the system time, $\mathbf{x}_U = [x\, y\, z]^T$ the user position at the time of reception t'_U according to the system time. All position vectors are referenced to an earth-centered earth-fixed (ECEF) coordinate system (CS). The distance the signal travels between the user and ith SV is then

$$\begin{aligned} \|\mathbf{x}_U - \mathbf{x}_{S,i}\| &= R_i && (1.1)\\ &= c\,(t'_U - t_{S,i}) && (1.2) \end{aligned}$$

where c is the speed of light. Since the receiver time is biased to the system time by an a proiri unknown shift $\delta t_U = t_U - t'_U$, the receiver can only measure so called *pseudorange* ρ_i to ith SV which is the true range R_i biased by the clock offset in meters $b = c \cdot \delta t_U$

$$\begin{aligned} \rho_i &= c \cdot (t_U - t_{S,i}) && (1.3)\\ &= c \cdot (t'_U - t_{S,i}) + c \cdot \delta t_U && (1.4)\\ &= R_i + b. && (1.5) \end{aligned}$$

Provided the receiver knows the system time of transmission of at least four SVs $\{t_{S,i}\}_{i=1}^{I}$, their positions at that time $\{\mathbf{x}_{S,i}\}_{i=1}^{I}$ and its own time t_U so that it can form the pseudoranges $\{\rho_i\}_{i=1}^{I}$ using (1.7), the user position $\mathbf{x}_U$ and clock bias b can

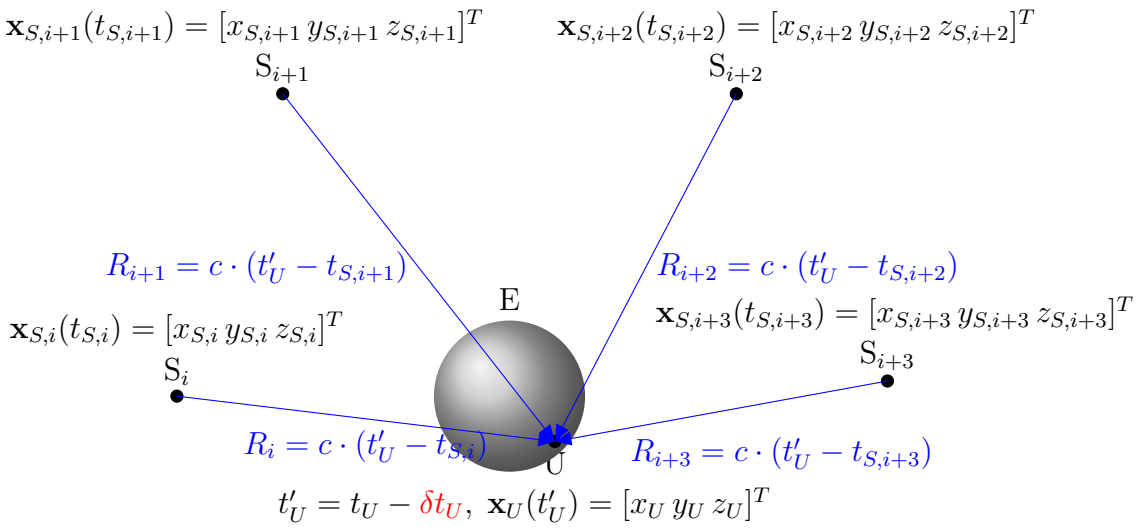


Figure 1.1: Position estimation using TOA measurements in GNSS

be estimated by solving the set of nonlinear equations for $i = 1, \ldots, I$

$$\sqrt{(x_U - x_{S,i})^2 + (y_U - y_{S,i})^2 + (z_U - z_{S,i})^2} + b = \rho_i \tag{1.6}$$

where I denotes the number of visible SVs. The situation is depicted in Figure 1.1. If $I > 4$, the solution of the nonlinear equations is said to be overdetermined. However, the measurements of pseudoranges are always embedded in some noise $w_{\rho,i}$ being a random variable

$$\rho_i = R_i + b + w_{\rho,i}. \tag{1.7}$$

Hence, one has to resort to the estimation theory [43] to optimally determine the estimates of user position $\hat{\mathbf{x}}_U = [\hat{x}_U\, \hat{y}_U\, \hat{z}_U]^T$ and the clock bias $\hat{b}$.

1.1.2 Velocity Estimation

The user velocity $\mathbf{v}_U = [\dot{x}_U\, \dot{y}_U\, \dot{z}_U]^T$ can be estimated from the measurements of the frequency shifts of the [1]incoming signals. The frequency shift of the incoming signal from ith SV $f_{s,i}$ can be modeled as a sum of the Doppler frequency shift $f_{d,i}$, [2]oscillator frequency offset $-\delta f$ and the known residual intermediate frequency (IF)

[1]Estimating user velocity from differences of pseudoranges is much less precise [12, 13].

[2]The negative sign is due to the difference of the input frequency and the frequency of the local oscillator when downconversion.

f_{IF} from the local oscillator

$$f_{s,i} = f_{d,i} - \delta f + f_{IF}. \tag{1.8}$$

Note that the range derivative $\dot{R}_i$ is related to the Doppler shift $f_{d,i}$ as

$$\dot{R}_i = -\frac{f_{d,i}}{f_{c,i}} \cdot c \tag{1.9}$$

where $f_{c,i}$ is the [3]carrier frequency of ith SV. As long as the same local oscillator (LO) is used for down conversion and local time measurement stamping so that the clock drift $\dot{b}$ is related to the oscillator frequency offset δf as

$$\dot{b} = \frac{\delta f}{f_{c,i}} \cdot c \tag{1.10}$$

the noiseless pseudorange first derivative, named as *pseudorange rate*, is related to the Doppler frequency as follows

$$\begin{aligned} \dot{\rho}_i &= \dot{R}_i + \dot{b} && (1.11)\\ &= -\frac{f_{d,i}}{f_{c,i}} \cdot c + \frac{\delta f}{f_{c,i}} \cdot c && (1.12)\\ &= -\frac{f_{s,i} - f_{IF}}{f_{c,i}} \cdot c. && (1.13) \end{aligned}$$

It should be noted that noiseless pseudorange rate and frequency shift without the known IF are related via a scaling constant. Taking the first derivative of the range, we get the relation of the range rate, the user velocity vector $\mathbf{v}_U$ and the velocity vector of ith SV $\mathbf{v}_{S,i} = [\dot{x}_{S,i}\, \dot{y}_{S,i}\, \dot{z}_{S,i}]^T$

$$\begin{aligned} \dot{R}_i &= \frac{\partial \left\| \mathbf{x}_U - \mathbf{x}_{S,i} \right\|}{\partial t} && (1.14)\\ &= -\mathbf{1}_i^T \cdot (\mathbf{v}_U - \mathbf{v}_{S,i}) && (1.15) \end{aligned}$$

where $\mathbf{1}_i$ is the unit line-of-sight vector between the user and the SV

$$\mathbf{1}_i = \left[\frac{x_U - x_{S,i}}{R_i} \; \frac{y_U - y_{S,i}}{R_i} \; \frac{z_U - z_{S,i}}{R_i} \right]^T. \tag{1.16}$$

[3]We index carrier frequencies for different SVs, since we consider more GNSS systems. However, even for one GNSS system, the carrier frequencies can vary over received signals - e.g. GLONASS uses frequency division multiple access (FDMA), or signals at multiple frequencies from different SVs can be combined.

Consider noisy frequency shift measurements and assume that the velocity vector of the SV at the time of transmission is known to the user. We can get the estimate of the user velocity $\hat{\mathbf{v}}_U$ and the estimate of the clock drift $\hat{\dot{b}}$ based on the following set of equations $i = 1, \ldots, I$

$$-\frac{f_{s,i} - f_{IF}}{f_{c,i}} \cdot c = -\mathbf{1}_i^T \cdot (\mathbf{v}_U - \mathbf{v}_{S,i}) + \dot{b} + w_{\dot{\rho},i} \tag{1.17}$$

where $w_{\dot{\rho},i}$ is a random variable representing the noise contribution. From this time on, we will consider the pseudorange rate being noisy from the frequency shift measurement so that

$$\dot{\rho}_i = -\mathbf{1}_i^T \cdot (\mathbf{v}_U - \mathbf{v}_{S,i}) + \dot{b} + w_{\dot{\rho},i}. \tag{1.18}$$

1.1.3 Channel Model

Let $s_i(t, \mathbf{q}) \in \mathbb{C}$ denote the complex envelope of the transmitted signal by ith SV at time t with encoded data message $\mathbf{q}$ denoted as a vector of channel symbols $\mathbf{q}_k \in \mathbb{R}^{N_q}$, $N_q \in \mathbb{N}$ generated at times kT_s where $k \in \mathbb{N}$ is the discrete time index and T_s is the period of channel symbols

$$\mathbf{q} = [\ldots \mathbf{q}_k \ldots]^T. \tag{1.19}$$

The signal propagates to the user that receives noisy baseband signal $y(t) \in \mathbb{C}$ being a composition of all the transmitted signals delayed in time by $\tau_i = t_U - t_{S,i}$, attenuated due to energy dissipation, atmospheric attenuation and other effects by a constant $\alpha_i \in \mathbb{R}$, shifted in frequency by $f_{s,i}$ due to the Doppler effect, residual IF and oscillator frequency offset, having unknown carrier phase offset $\varphi_i \in \mathbb{R}$, and is embedded in noise $n(t) \in \mathbb{C}$

$$y(t) = \sum_{i=1}^{I} \alpha_i \exp\left(\mathrm{j}\left(2\pi f_{s,i} t + \varphi_i\right)\right) s_i\left(t - \tau_i, \mathbf{q}\right) + n(t). \tag{1.20}$$

The noise $n(t)$ can be successfully modeled as a band-limited complex white Gaussian noise (WGN) with single sided power spectral density (PSD)[4]

$$\mathcal{S}_n(f) = \begin{cases} 2N_0, & |f| < B_R \\ 0, & \text{else.} \end{cases}$$

where f is the frequency, $N_0 \in \mathbb{R}$ and B_R is the receiver filter bandwidth. As the geometry of the user and the SVs evolves over time, parameters τ_i, $f_{s,i}$, φ_i,

[4]Not considering interference.

α_i, N_0 change. GNSS systems are designed for the situation where these parameters change slowly with respect to T_s, being abbreviated as *slowly time-varying parameters* (STVP). Such a channel is named as *linear additive WGN channel with STVP* and synchronization techniques can be employed [53, 54] to either estimate or eliminate the channel parameters to get the estimate of the transmitted data. Since we are interested in estimating the signal delay τ_i and frequency shift $f_{s,i}$, elimination of φ_i, α_i, N_0 may be adopted. In practice, the elimination takes place at the early stage of the receiver synchronization to simplify the process of obtaining coarse estimates. Parameters α_i, N_0 are then needed as a signal quality measure for optimizing the PVT estimation and control of the synchronization process. Carrier phase tracking of $\phi_i = 2\pi f_{s,i} t + \varphi_i$ can increase the probability of correct data detection using soft decision algorithms and can be incorporated to the PVT estimation process with additional reference receiver to get local precise PVT estimates [12, 13, 50] or can be used stand-alone at long observations [56, 57].

1.1.4 Signal Structure

GNSS signals are designed as *direct-sequence spread-spectrum* (DS SS) modulated signals with code or frequency division multiple access (CDMA, FDMA) to differentiate the SVs of a system [50, 51]. The typical modulations are the *binary-phase-shift keying* (BPSK) or *binary-offset keying* (BOC) [13, 58–60], in some cases with minor modifications such as separate *pilot* and *data signals* in the inphase or quadrature components, or multiplexed in time (GPS L1A, L2C), code (Galileo E1) or frequency (GLONASS L1/L2). The most challenging signal is the alternate BOC (AltBOC) modulated signal transmitted at Galileo E5 band with reference bandwidth of 50 MHz, offering decimeter level precision of delay tracking [5, 41, 59, 61–63].

The primary reasons for selection of these signals are the sharpness of the signal autocorrelation function ensuring high precision of the delay tracking, suppression of the distanced reflected signals, resistance to both narrowband and wideband interference, short occasional signal fadings or ionospheric scintillation [13]. The pilot signals can be used to increase receiver sensitivity or precision thanks to the possibility of using discriminators that are sensitive to data bit transitions and the possibility of long coherent integration times. However, the transmitted power must be shared between the data and pilot signals, therefore the probability of correct data detection is lowered. If the signal power is as low as synchronization of the data signal cannot be maintained, but the synchronization of the pilot signal is still possible, the benefit of the pilot signal is that no reacquisition is needed to restore the data signal tracking. The GNSS signals are always transmitted with *constant envelope* to fully utilize the

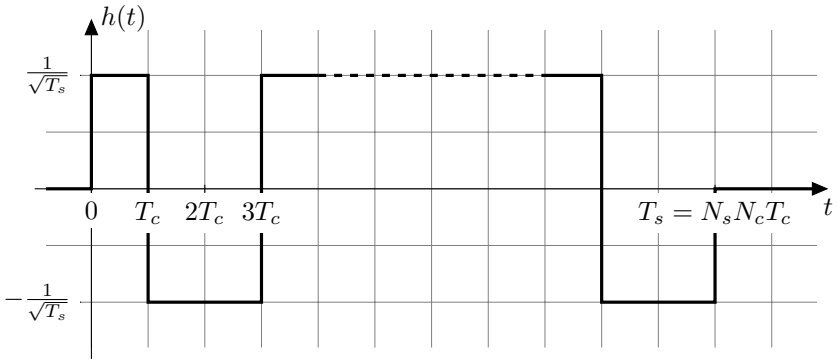


Figure 1.2: Modulation impulse of GPS L1 C/A code signal. Symbol T_c denotes chip period, symbol N_c the number of chips within a code period, symbol N_s the number of code periods per a channel symbol.

power available on board the SVs.

Any GNSS signal, or its data or pilot component, can be generally described as a special case of a *linear multi-dimensional modulation* (LMDM) [37, 45]

$$s(t) = A \sum_k \mathbf{q}_k^T \mathbf{h}\left(t - kT_s\right) \tag{1.21}$$

where A is the signal amplitude, $\mathbf{q}_k$ is a vector of channel symbols and $\mathbf{h}(t) \in \mathbb{C}^{N_q}$ is a vector of modulation impulses that are essentially nonzero for $t \in \langle kT_s, (k+1)T_s)$ where T_s is the period of the channel symbol. For GPS L1 C/A signal, the channel symbol is scalar $\mathbf{q}_k = [q_{k,1}]^T$ and contains the Hamming-encoded data bit with period $T_s = 20$ ms. The modulation impulse is also scalar containing 20 periods of the Gold code. The amplitude of the modulation impulse is normalized so that the energy of $h(t)$ is unit. The situation is depicted in Figure 1.2. Examples how to represent BOC, AltBOC, inphase and quadrature signals are given in [45]. The advantage of such description is its universality to derive common signal processing algorithms or performance characteristics using either theory or simulations.

1.1.5 Signal Conditioning

Before the received signal is processed by the digital signal processors (DSP), it must be first conditioned and compromise must be made about the digitized signal quality, receiver size, costs and power consumption. The *antenna* should be of a hemispheric radiation pattern so that it does not receive ground-reflected signals and amplifies

the direct signals. The signal should then be preamplified by a *low noise amplifier* (LNA) as soon as possible to get the lowest possible noise figure, and then *coarsely filtered*. However, filtering before LNA may help to prevent the LNA from being overdriven by strong signals within the antenna and LNA operation bandwidth, but will likely increase the overall noise figure due to its insertion losses. On condition that the bandwidth of the signal that LNA should amplify is large, the LNA will have to be of high dynamic range with high power consumption. Strong signals within the band will easily overdrive the amplifier. Hence, compromises must be found for various environments and signal spectrum allocation.

The signal is then *downconverted* and *filtered* with a narrowband filter. One or two other amplifiers are cascaded to the signal path with at least one of variable gain for *automatic gain control* (AGC) in order to optimally fit the input signal into the range of the *analog-to-digital converter* (ADC). One of the amplifiers has typically high gain (e.g. 75 dB) since the input signal is buried in noise. For down conversion, superheterodyn receivers or direct conversion concepts are typically used. The former typically splits the down conversion into two steps to increase the receiver selectivity and sampling is performed at higher Nyquist zone. The latter is not fitted to a particular frequency plan and is thus more suitable for reception of signals at various frequencies thanks to the existence of no mirror frequencies.

The stability of the *local oscillator* is another crucial parameter in GNSS receivers. It determines the frequency range of the signal acquisition and consequently the *time to first fix* (TTFF), performance of PVT filtering algorithm, the time range for which warm start may be performed since the last position fix and others. Since GNSS is based on TOA measurements, the local oscillator is said to be *the heart of the receiver*. It is a must in GNSS receivers to derive all the clock signals and mixing frequencies from a common LO so that the number of unknown parameters for the PVT estimation/filtering is kept minimal.

The *sampling frequency* of receivers that receive only the inphase component should be always larger than twice the filtered signal bandwidth, for receivers that process both inphase and quadrature components, it should be at least the filtered bandwidth of the signal. The benefit of using solely the inphase component for e.g. GPS L1 C/A is the lower complexity of the signal processing, since complex multiplication involves four real multiplications and two real additions. However, the sampling frequency must be doubled. Larger filter bandwidth and sampling frequencies challenge the design of DSPs, but enable implementation of multipath mitigation techniques such as narrow correlator [64], multipath-estimating delay-locked loop (MEDLL) [65] or other multipath techniques based on direct positioning [66, 67].

It has been proved that using 1-bit signal representation introduces only 1.96 dB

degradation, 2-bit representation 0.55 dB and 3-bit representation 0.16 dB degradation of the code tracking precision [13]. The advantage of using higher bit *quantization* is the ability to better mitigate the interference and jamming effects. Some receivers use techniques to mitigate the effect of narrowband interference by measuring the statistics of the sampled signal - J/N meter [68] or using a bank of digital filters. The necessary condition for such interference mitigation is the linear functionality of the RF chain and prompt AGC reaction to the increasing signal power.

1.1.6 Modularization of Traditional GNSS Receiver

As long as the input signal is properly conditioned, it can be further processed to obtain the PVT estimates. If the signal parameters τ_i, $f_{s,i}$, φ_i, α_i, N_0 are STVP in a linear additive WGN channel, the signal samples $y_n = y(nT_p)$ are sufficient statistics for the estimation of these parameters and the transmitted data symbols [53, 54] meaning that no information is lost by sampling. Symbol T_p denotes the sampling period. All the signal processing is done in discrete time.

In a traditional receiver, the overall signal processing steps are modularized as follows, making a solution with acceptable complexity and feasible in real time,

- signal acquisition
- signal tracking and bit synchronization
- frame synchronization, data demodulation and decoding
- navigation data handling and pseudorange formation
- PVT estimation/filtering.

1.1.6.1 Signal Acquisition

In order to track the channel parameters of the input signal and get the estimates of the data symbols, coarse estimates of signal delay τ_i and frequency shift $f_{s,i}$ must be obtained[5]. The process is named as signal *acquisition*[6]. The nature of DS SS signals implies that search techniques must be used [51]. The most common techniques based on maximum likelihood (ML) criterion include *serial search* where the signal replica

[5]Strictly speaking, we estimate the fractional part of the code delay due to the periodic nature of the pseudorandom codes. We will come to this point when discussing the pseudorange formation algorithm.

[6]In fact, signal acquisition is a process of both signal detection and channel parameter estimation.

is periodically shifted in time, correlated with the incoming signal and threshold crossing is detected, *parallel search* where the input signal is correlated with all sufficient number of replicas at the same time and the highest value is compared with the threshold, and *hybrid search* which is a linear scale between the serial and the parallel search. These techniques can be implemented using the DSP correlators.

However, parallel search techniques based on fast Fourier transform (FFT) such as *parallel code space search* (PCSS) and *double block zero padding* (DBZP) are becoming dominant thanks to their lower complexity, availability of FFT libraries for DSPs, CPUs, GPUs, ASICs or FPGAs. When implementation of the FFT-based acquisition methods, the signal delay τ_i must be extrapolated to the time of transition to the tracking. If the transition time is high, the user-to-SV pseudorange rate may change and a serial search in significantly smaller delay range is usually adopted to align the replica with the incoming signal.

An interesting method based on iterative message passing on a cycle factor graph has been proposed to efficiently acquire pseudorandom sequences of long codes [69–71] and adopted to GNSS [72]. The method is an analogy to decoding of low-density-parity-check (LDPC) codes in digital communications, but is strictly dependent on the spreading code structure. For codes with short cycles in their graphical representation, the method suffers from sensitivity degradation. In case of GPS L1 C/A Gold codes, the sensitivity degradation is about 7 dB compared to the maximum likelihood method based on parallel search.

1.1.6.2 Signal Tracking and Bit Synchronization

The signal tracking in a traditional GNSS receiver is accomplished using code and carrier tracking loops - *delay-locked loop* (DLL), *frequency-locked loop* (FLL) and *phase-locked loop* (PLL). These loops are an implementation of an iterative ML estimator of the code delay τ_i, frequency shift $f_{s,i}$, and carrier phase offset φ_i, respectively [53, 54]. A separate tracking loop can be analytically described as a feed-back system (FBS) with its equivalent model and its basic characteristics such as discriminator characteristic, discriminator gain, equivalent loop noise autocorrelation function, loop filter order and its bandwidth. An implementation of the FBS setup differs for various receiver stages. Right after the acquisition, DLL and FLL with relatively wide equivalent loop noise bandwidth are employed in order to promptly align the replica with the incoming signal and enter the steady state. Afterwards, the bandwidth is lowered to filter out the noise. To improve precision, the tracked frequency from the carrier tracking loop is often fed to the code tracking loop of the same SV channel and multiplied by a scaling constant to remove all the dynamics

from the code tracking loop and lower the bandwidth down to 0.1 Hz. The carrier measurements are less noisy that the those on code. The FLL can further assist PLL to track the dynamics of the carrier phase so that the PLL bandwidth can be also lowered.

The derivation of the iterative ML estimate of the STVP channel parameters introduces sufficient statistics in the form of the cross-correlation between the received signal and its replica or the first derivative of its replica. Practical implementation is an approximation of the replica derivative by a difference of the *early/late* (E/L) replicas [53,54]. The time difference between the E/L replicas $2\Delta\tau_s$ and the coherent integration time T_I are the crucial parameters influencing the analogue bandwidth of the FBS. The correlation and replica generation are the most computationally demanding operations, the sampling frequency and the number of quantization bits should hence be considered with care.

Common DLL discriminators are the *normalized early-minus-late power discriminator* and the *normalized dot-product discriminator* since they are insensitive to a data bit transition and carrier phase offset. Normalization is employed to eliminate the unknown signal amplitude influencing the overall FBS gain and consequently its bandwidth. The drawback of the discriminators insensitive to the bit transition is that they suffer from squaring loss which can be minimized by increasing the coherent integration time. Code discriminators relying on the carrier phase estimate are not common in GNSS receivers, since the PLL has worse sensitivity than the DLL. A cycle slip in PLL would then result in a loss of lock of the DLL. The FLL discriminators are based on the difference between the current phase shift estimate and its previous value, thus approximating the first derivative of the carrier phase. They can be either sensitive to the data bit transition, but must be employed only between the intervals of the data bit transition, or insensitive to data bit transition and used at all times. The latter is a common choice for signals with secondary code where the data bit transition may occur every primary code period. Although higher integration time lowers the squaring loss of the DLL tracking jitter, it lowers the FLL pull-in range. In practice, FLL operates at lower integration time because of that reason. The PLL discriminators should be always insensitive to the data bit transition, except for those tracking pilot signals with secondary code in synchronization. Commonly, the *atan discriminator* for data signals and *atan2 discriminator* for pilot signals are used due to their linear characteristics and independence of C/N_0 over the operating range.

It should be noted that a design of the tracking loops is a compromise respecting user dynamics, requirements on sensitivity, tracking jitter, mean time to synchronization failure and others. For high dynamic applications, FLL-aided DLLs are desired

with FLL having higher order loop filters so that they do not produce systematic errors of the tracking jitter caused by higher derivatives of the delay. In case of ionospheric-free combinations, the DLL tracking jitter should be minimized not to degrade the advantage of such compensation.

Before data demodulation and decoding, the data bit transitions must be continuously localized within the flow of the correlator output values. A common approach for the signal without secondary code is to create a *histogram of carrier phase reversals* and find that with the highest count. The histogram may serve also as an indicator of loss of lock if there is no bin with dominant count. For modernized GNSS signals with secondary codes, it is much more straightforward to find the bit boundaries in a different way assuming carrier phase in lock. For every new correlator output value, binary hypothesis if the bit boundary is present or not is tested using a *Neyman-Pearson detector* that correlates the secondary code with the in-phase correlator output values and decides according to the threshold crossing. To improve the probability of detection, results for more secondary code periods can be noncoherently combined. No threshold crossing is an indicator of loss of lock.

To measure the signal quality, the quantity signal power $C_i = \frac{1}{2}\mathrm{AvE}\left[\left|\alpha_i s_i\left(t\right)\right|^2\right]$ to noise PSD N_0, abbreviated as C/N_0, is estimated in GNSS receivers. There are two techniques commonly used to estimate C/N_0. One dedicates a single correlator to correlate the input signal with a replica that is not certainly contained in the input signal and estimates thus the noise PSD N_0. The signal power is estimated based on the correlator output values being linearly dependent upon the signal amplitude at high C/N_0. Another approach uses a combination of several correlator output values to estimate both C and N_0. The variance of the correlator output values is in direct relation to N_0.

1.1.6.3 Frame Synchronization, Data Demodulation and Decoding

The transmitted data of older GNSS signals (GPS L1 C/A, GLONASS L1/L2) are protected by relatively short *Hamming codes* to correct one-bit error and detect two-bit error in about 30-bit words. Hard estimates of transmitted data bits are produced based on a simple decision of the bit reversal. A *synchronization pattern* is periodically inserted into the transmitted data symbols to successfully detect the beginning of a frame. A Neyman-Pearson detector can be used to decide about the presence of the frame start. Parity of the buffered symbols are then checked for data words within the frame.

Modernized GNSS signals (GPS L2C, GPS L5C, EGNOS, WAAS, Galileo INAV, FNAV) are first protected with a *CRC* over the whole frame, then *convolutionally*

encoded, optionally *interleaved* (Galileo signals) and finally extended with a *synchronization pattern* at the beginning of the frame. The convolution code significantly lowers the bit error ratio (BER), and the interleaving protects the signal against fadings. All of these signals adopt the same convolution encoder with constrain length $K = 7$ and identical CRC encoder of 24 bits. The difference is in the encoded signal size, presence of tail bits and bit inversion of one of the branch in the convolution encoder. To decode convolutionally encoded signals, soft decision algorithms should be adopted such as Viterbi algorithm taking the inphase[7] correlator outputs as a metric.

GPS L1C signal is planned to incorporate an *LDPC code* featuring optimized BER. The decoding will be based on iterative message passing techniques on a factor graph representing the encoding matrix.

1.1.6.4 Navigation Data Handling and Pseudorange Formation

Successfully decoded navigation data bits are stored into a *navigation data storage*. The navigation data storage must be designed to carefully check data validity and consistency. This, more programming than mathematical, task is in practice implemented using separate data structures for the incoming data and for the stored and consistent data being used for PVT estimation. For SBAS systems, the old data structures might be needed because the SBAS corrections cannot apply to the new data due to the law of causality.

The GNSS SVs transmit both data corresponding to the SV and data corresponding to the system or corrections to other systems. The SV data include ephemeris, clock corrections, health, and alert flags for integrity purposes. The data common to the system include almanac and health of other SVs, ionospheric corrections (by a global map), time conversion parameters, and others. Every decoded frame entity contains the time of transmission of the first bit in the frame. The frames are transmitted synchronously from all the SVs according to the local time scale, but the actual times differ due to the biases of the onboard oscillators.

The navigation data storage should also provide functions for calculation of the PVT of a SV for a given time of transmission, functions for time, ionospheric, tropospheric corrections and functions for conversion between various coordinate systems.

In order to calculate the *time of transmission* of ith SV $t_{S,i}$, the following algorithm can be used. When a time mark corresponding to the beginning of the frame

[7]Strictly speaking, the PLL must be locked and the correlator output values should be also correlated with the secondary code. These values can then be used as a signal metric.

is detected, this time[8] is remembered t_F and an accumulator starts counting the number of elapsed data bits and their fractional part up to the time epoch of PVT estimation. The number of bits $N_{b,i}$ are available from the bit synchronization process and its fractional part $\Delta N_{b,i}$ is known from the code NCO accumulators. The time of transmission then equals

$$t_{S,i} = t_F + (N_{b,i} + \Delta N_{b,i})\, T_b \tag{1.22}$$

where T_b is the bit period. For histogram-based bit synchronization, one bit delay is introduced. The *user time* t_U must be also related to the system time scale. After the receiver is switched on after a long time, this time might be significantly biased and can be initialized as

$$t_U = t_{S,i'} + R_{min}/c \tag{1.23}$$

where $i' = \text{argmax}_i \{t_{S,i}\}_{i=1}^{I}$ and R_{min} is the distance between a zenith SV and the user on the Earth's surface which is a constant. The pseudorange can then be formed according to (1.7).

1.1.6.5 PVT Estimation/Filtering

As the pseudoranges and pseudorange rates are formed, the positions and velocities of the SVs are calculated, user PVT can be estimated. The simplest practical method is the *least squares* (LS) linearized about the last position fix. For the first estimation, the initial position is selected as the center of the Earth. On condition that the user is on Earth or in the atmosphere, this initial condition along with the user time initialization from Section 1.1.6.4 always converges for all GNSS constellations [13].

Since some of the pseudoranges and pseudorange rates are of better quality than others, C/N_0 threshold can be applied to exclude noisy measurements. Next, satellites with low elevation angles suffer from severe multipath and ionospheric refraction difficult to remove without dual frequency measurement. Elevation mask of 5° to reject low elevation satellites from PVT measurement set is a standard practice. If the geometric diversity of the visible constellation is poor so that the position estimation precision would be much worse than pseudorange standard deviation, the PVT is not estimated at all. Typically, position-dilution-of-precision (PDOP) threshold equal six is used in GNSS receivers.

To improve the performance of the PVT estimation, *weighting* can be incorporated to the least squares criterion (WLS). A straightforward approach is to construct the weights based on C/N_0 estimates. However, these estimates reflect more how

[8] This time is called z-count for GPS signals.

the pseudoranges are noisy, but other significant sources of errors such as ionospheric errors after correction for single frequency receivers (0-7.5 m), multipath error (1 m), ephemeris and clock prediction errors (1.5 m), become dominant if the tracking loops are properly designed. Weighting using elevation angles is suggested because of these reasons.

A priori knowledge about user dynamics can be embodied into the estimation process. Typically, the *extended Kalman filter* (EKF) is used with various Gauss-Markov models of the user PVT. In airborne application, even acceleration models are used. The measurement covariance matrices should be constructed respecting also other errors than noise to get the best filtered values. For velocity and higher order motion models, a large oscillator drift rate can limit the filtering performance.

For safety-of-live (SoL) applications, it is desirable to identify if any of the measurements are much more inaccurate than expected and can significantly bias the PVT estimates. A technique named *receiver autonomous integrity monitoring* (RAIM) estimates the PVT excluding one measurement of a SV and remembers the result. The same is repeated for other SVs. If one of these PVT estimates is distanced by an unexpected amount to the others, it is excluded from the PVT estimation process. To lower the complexity, QR factorization can be used to get the statistics for all the SVs at once.

1.2 Receiver Architectures

In this section, we discuss various receiver architectures - conventional architecture, vector tracking architecture, and direct positioning architecture. The attention will be paid to construction of measurement equations for these architectures, cooperation of the tracking loops with the PVT estimator and the consequent performance aspects.

Let us first define the following vectors: a vector of signal samples $\mathbf{y} = [y_0 \dots y_{K-1}]^T$, the vector of pseudoranges $\boldsymbol{\rho} = [\rho_1 \dots \rho_I]^T$, the vector of pseudorange rates $\dot{\boldsymbol{\rho}} = [\dot{\rho}_1 \dots \dot{\rho}_I]^T$, and the PVT vector

$$\boldsymbol{\gamma} = \begin{bmatrix} \mathbf{x}_U \\ b \\ \mathbf{v}_U \\ \dot{b} \end{bmatrix}. \tag{1.24}$$

We will generally denote an estimate of random variable a at time k with hat $\hat{a}$, prediction with tilde $\tilde{a}$. We further distinguish noiseless pseudoranges values with

asterisk a^*. For example, the noiseless pseudorange to ith SV ρ_i^* equals

$$\rho_i^* = \rho_i - w_{\rho,i} \tag{1.25}$$

the noiseless pseudorange rate is then

$$\dot{\rho}_i^* = \dot{\rho}_i - w_{\dot{\rho},i} \tag{1.26}$$

1.2.1 Conventional Architecture

The conventional architecture comprises *local tracking channels* for each SV providing pseudoranges and pseudorange rates to be used as observables by the *navigation processor* estimating the user PVT, see Figure 1.3(a). Each local tracking channel estimates the parameters separately and independently based on the NCO accumulator states of the code and carrier tracking loops and pseudorange formation block using time marks from the navigation data. Since the estimation process is split between two steps, it is often called as *two-step approach*.

1.2.1.1 Measurement Equation for Estimation of Pseudoranges and Pseudorange Rates

In a conventional receiver, the pseudoranges $\{\rho_i\}_{i=1}^I$ and pseudorange rates $\{\dot{\rho}_i\}_{i=1}^I$ are first calculated taking the received signal samples $\mathbf{y} = [y_0 \dots y_{K-1}]^T$ as a measurement for $k = 0, \dots, K-1$

$$y_k = \sum_{i=1}^{I} \alpha_i \exp\left(\mathrm{j}\left(2\pi\left(-\frac{\dot{\rho}_i^*}{c} f_{c,i} + f_{IF}\right) kT_p + \varphi_i\right)\right) s_i\left(kT_p - \rho_i^*/c\right) + n_k. \tag{1.27}$$

Let us assume that $T_p = 1/B_R$, the noise component $n_k = n(kT_p)$ can then be modeled as complex WGN with single sided PSD $2N_0/T_p$ for $k = 0, \dots, K-1$. Let $\nu_{i,k}\left(\rho_i^*, \dot{\rho}_i^*\right)$ be a time-dependent function that shifts the transmitted signal in time $\tau_i = \rho_i^*/c$ and in frequency by $f_{s,i} = (-\dot{\rho}_i^* f_{c,i}/c + f_{IF})$ so that

$$\nu_{i,k}\left(\rho_i^*, \dot{\rho}_i^*\right) = \exp\left(2\pi\mathrm{j}\left(-\frac{\dot{\rho}_i^*}{c} f_{c,i} + f_{IF}\right) kT_p\right) s_i\left(kT_p - \rho_i^*/c\right) \tag{1.28}$$

and let $\epsilon\left(\alpha_i, \varphi_i\right)$ be an exponential with phase φ_i and scaling α_i. The received signal can then be expressed as

$$y_k = \sum_i \epsilon\left(\alpha_i, \varphi_i\right) \nu_{i,k}\left(\rho_i^*, \dot{\rho}_i^*\right) + n_k. \tag{1.29}$$

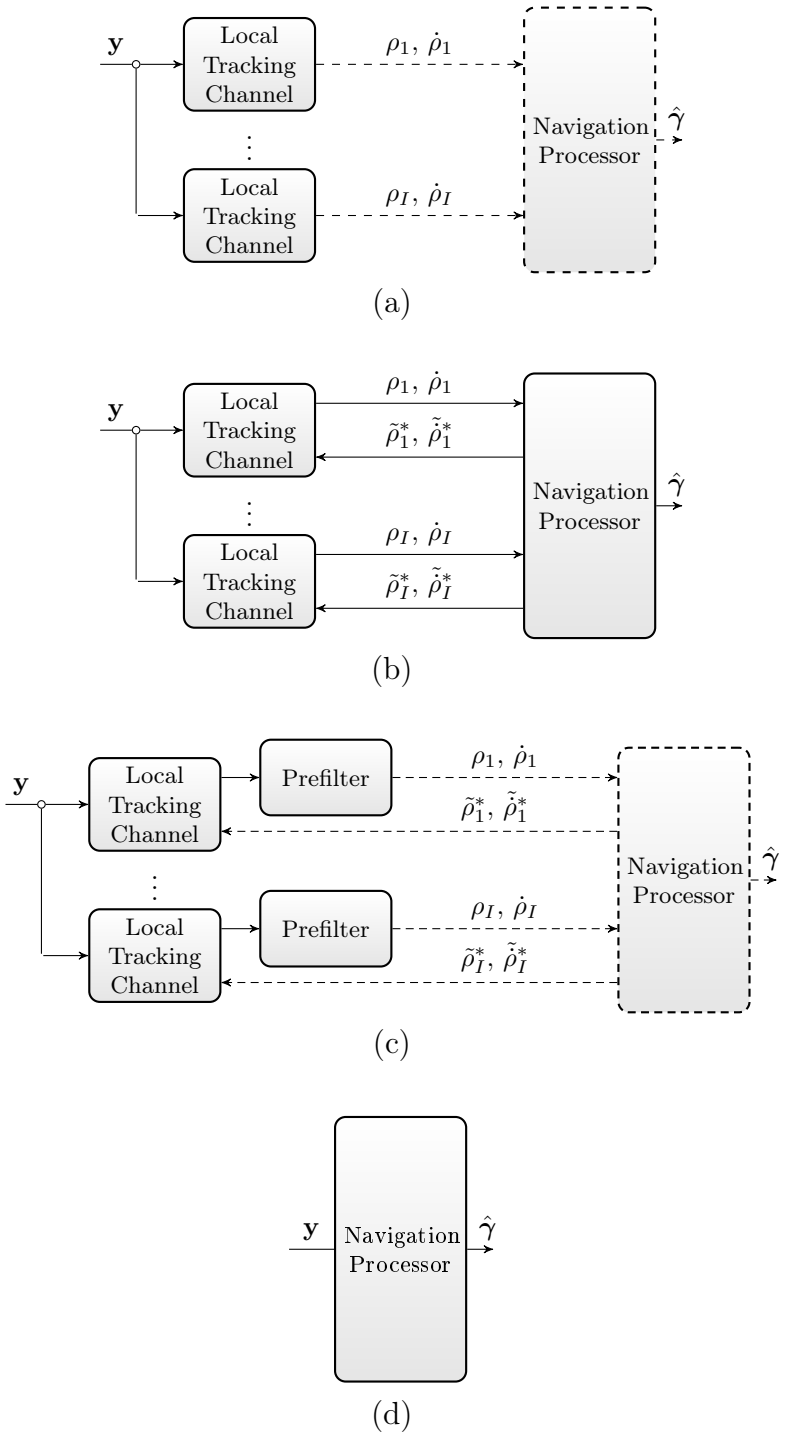


Figure 1.3: Receiver architectures: (a) conventional architecture, (b) vector tracking architecture, (c) vector tracking architecture with prefilters, (d) direct positioning architecture. Lines denoted as dash operate at low rate (navigation update rate - typically 0.1 s or 1 s), whereas solid lines operate at much higher rate.

Defining a vector of complex WGN noise $\mathbf{n} = [n_0 \dots n_{K-1}]^T$ and a vector of shifted signal $\boldsymbol{\nu}_i(\rho_i^*, \dot{\rho}_i^*) = [\nu_{i,0} \dots \nu_{i,K-1}]^T$, a set of (1.29) can be rewritten as, forming the measurement equation for estimation of noiseless pseudoranges and pseudorange rates,

$$\mathbf{y} = \sum_i \epsilon(\alpha_i, \varphi_i)\, \boldsymbol{\nu}_i(\rho_i^*, \dot{\rho}_i^*) + \mathbf{n}. \tag{1.30}$$

1.2.1.2 Measurement Equation for PVT Estimation

Substituting (1.1) into (1.7), we can rewrite the equation for pseudorange measured to ith SV as

$$\rho_i = \|\mathbf{x}_U - \mathbf{x}_{S,i}\| + b + w_{\rho,i}. \tag{1.31}$$

Let us now define a function $g_{\mathbf{x},i}(\mathbf{x}_U, b) = \|\mathbf{x}_U - \mathbf{x}_{S,i}\| + b = \rho_i^*$ that calculates the noiseless pseudorange to ith SV from the user position $\mathbf{x}_U$, let $\mathbf{g}_\mathbf{x} = [g_{\mathbf{x},1} \dots g_{\mathbf{x},I}]^T$ be a concatenated vector function of $g_{\mathbf{x},i}$. Defining a vector of pseudoranges $\boldsymbol{\rho} = [\rho_1 \dots \rho_I]^T$, the measurement equation for the estimation of user position $\mathbf{x}_U$ and clock bias b can be formed as

$$\boldsymbol{\rho} = \mathbf{g}_\mathbf{x}(\mathbf{x}_U, b) + \mathbf{w}_{\boldsymbol{\rho}} \tag{1.32}$$

where $\mathbf{w}_{\boldsymbol{\rho}} = [w_{\rho,1} \dots w_{\rho,I}]^T$ is an additive noise vector.

Similar steps can be applied to derive the measurement equation for estimation of user velocity $\mathbf{v}_U$ and clock drift $\dot{b}$. Let us define function $g_{\mathbf{v},i}(\mathbf{v}_U) = -\mathbf{1}_i^T \cdot (\mathbf{v}_U - \mathbf{v}_{S,i}) + \dot{b} = \dot{\rho}_i^*$ that calculates the noiseless pseudorange rate $\dot{\rho}_i^*$ to ith SV so that (1.18) can be rewritten as

$$\dot{\rho}_i = g_{\mathbf{v},i}\left(\mathbf{v}_U, \dot{b}\right) + \dot{w}_{\dot{\rho},i}. \tag{1.33}$$

Let $\mathbf{g}_\mathbf{v}\left(\mathbf{v}_U, \dot{b}\right) = [g_{\mathbf{v},1} \dots g_{\mathbf{v},I}]^T$ be a concatenated vector function of $g_{\mathbf{v},i}$. Defining a vector of pseudorange rates $\dot{\boldsymbol{\rho}} = [\dot{\rho}_1 \dots \dot{\rho}_I]^T$, a measurement equation for the estimation of user position $\mathbf{v}_U$ and clock bias $\dot{b}$ can be formed as

$$\dot{\boldsymbol{\rho}} = \mathbf{g}_\mathbf{v}\left(\mathbf{v}_U, \dot{b}\right) + \mathbf{w}_{\dot{\boldsymbol{\rho}}} \tag{1.34}$$

where $\mathbf{w}_{\dot{\boldsymbol{\rho}}} = [w_{\dot{\rho},1} \dots w_{\dot{\rho},I}]^T$ is an additive noise vector.

1.2.1.3 Estimator Decomposition

The estimation process of the PVT is decomposed into two steps - estimation of noiseless pseudoranges $\{\rho_i^*\}_{i=1}^I$ and pseudorange rates $\{\dot{\rho}_i^*\}_{i=1}^I$ separately and independently for each SV not considering signals from other SVs, since they are almost orthogonal due to CDMA. The considered measurement equation for ith SV from (1.30) simplifies to

$$\mathbf{y} = \epsilon\left(\alpha_i,\, \varphi_i\right) \boldsymbol{\nu}_i\left(\rho_i^*,\, \dot{\rho}_i^*\right) + \mathbf{n}. \tag{1.35}$$

The estimates of noiseless pseudoranges and pseudorange rates, being the pseudoranges and pseudorange rates for $i = 1, \ldots, I$, respectively,

$$\hat{\rho}_i^* = \rho_i \tag{1.36}$$
$$\hat{\dot{\rho}}_i^* = \dot{\rho}_i \tag{1.37}$$

are then used as observations for the PVT estimation in (1.32), (1.34).

1.2.2 Vector Tracking Architecture

The fact that the tracking loops and the PVT estimator operate *decoupled* is the most straightforward way of implementation being feasible with current technology and education of GNSS engineers. However, a concept of suboptimal cooperation between the tracking loops and the PVT estimator yielding increased sensitivity, named as *vector tracking*, has been proposed [73–76]. After the first decoupled PVT estimate, the receiver is switched to an architecture where the PVT estimates or predictions to the next epoch are used to drive the NCOs of the tracking loops, thus closing a global feed back. The architecture is depicted in Figure 1.3(b). The main benefit of such architecture is that the satellite channels of the receiver are controlled by the PVT estimator and are hence more resistant to signal fadings and can operate under much lower C/N_0, since the channels automatically support one another.

The main drawback that the PVT estimator has to operate at the same rate as the sampled outputs of the correlators has been removed by so called *prefilters* filtering the discriminator outputs [55,77–79], see Figure 1.3(c). Common approaches are the LS and KF combining both code and carrier measurements [55, 80, 81]. It was demonstrated that 7 dB sensitivity gain [55, 80, 82, 83] can be obtained because of this global feed back in a GPS L1 C/A receiver and 3 dB are due to the prefilters. In [84], it was shown that the vector tracking architecture is more resistant to interference due to the coupled channels. A theoretical framework for performance analysis has been opened in [83,85]. Since the channels of higher signal power support the low power channels, a C/N_0 estimator able provide unbiased estimates at tracking

threshold is needed. A suitable estimator using solely the correlator output values has been proposed in [80]. In practice, the channels under a certain C/N_0 threshold are rejected for PVT estimation in order to eliminate the bias of the C/N_0 estimator. Low power channels contribute negligibly to the overall PVT estimate, anyway, even if they are properly weighted or filtered. A phenomenon that biases of the pseudorange and pseudorange rate measurements contribute negatively to the other channels has been identified and has been investigated yet. This would be the case e.g. for severe multipath.

The measurement equation for the vector tracking architecture remain unchanged from those for conventional receiver in (1.30), (1.32), (1.34). However, the method how the pseudoranges and pseudorange rates are estimated changes. The method is not ML iterative anymore - no FBSs are present. The predicted values driving the replica NCOs become a linearizing point for the discriminators. The fact that the measurement equations are unchanged is reflected in no accuracy improvement at high C/N_0 operation compared to the conventional architecture. On the other hand, the sensitivity gain stems from a stable linearizing point for discriminators, which outputs are then filtered by the prefilters.

Despite the fact that the promising vector tracking architecture was introduced in early eighties, it has never been implemented in a commercial receiver. The reason for this is often said to be a lack of engineers skilled in both signal tracking and navigation. Parallel with digital communications, the cooperation of individual receiver layers is expected to boom in the very near future.

1.2.3 Direct Positioning Architecture

The division between the tracking channels and the PVT estimator can be removed and a single estimator can be derived. The measurement equation can be obtained by substituting for the noiseless pseudoranges and pseudorange rates in the measurement equation (1.30)

$$\mathbf{y} = \sum_{i=1}^{I} \epsilon\left(\alpha_i,\, \varphi_i\right) \boldsymbol{\nu}_i\left(\rho_i^*,\, \dot{\rho}_i^*\right) + \mathbf{n} \tag{1.38}$$

$$= \sum_{i=1}^{I} \epsilon\left(\alpha_i,\, \varphi_i\right) \boldsymbol{\nu}_i\left(g_{\mathbf{x}}\left(\mathbf{x}_U,\, b\right),\, g_{\mathbf{v}}\left(\mathbf{v}_U,\, \dot{b}\right)\right) + \mathbf{n}. \tag{1.39}$$

Estimation of user position $\hat{\mathbf{x}}_U$, clock bias $\hat{b}$, user velocity $\hat{\mathbf{v}}_U$, and clock drift $\hat{\dot{b}}$ from the vector of signal samples $\mathbf{y}$ is named as *direct position estimation* (DPE). The

architecture is depicted in Figure 1.3(d).

In [86–88], fundamental CRLB for direct positioning was derived. It is shown that the DPE performs better than the conventional architecture when some of the satellite signals are at low power. For the same C/N_0 ratios of all satellites, both CRLBs for position estimation are nearly identical. Direct positioning estimators based on WLS, ML, and particle filters have been proposed in [66, 86, 89, 90]. It was shown that there is a large improvement of multipath mitigation/estimation when using DPE approach compared to the conventional architecture.

The implementation of the DPE is suitable for pure SDR receivers, because several iterations on the same large data vectors are needed which could not be done with real-time correlators. The overhead of this method is large. So far these methods have been implemented in SDR receivers working on powerful PCs with multi-core CPUs [66] in postprocessing mode.

1.3 Receiver Hardware Concepts

Nowadays, two hardware concepts are employed in GNSS receivers - *traditional concept with real-time hardware correlators* and *software-defined-radio concept*. The former is a practical solution of commercial receivers reducing size, power consumption and costs, whereas the latter is at the time of writing a solution for researchers and special applications. In this section, we discuss these two concepts in more detail, looking mostly on how they can accommodate various receiver architectures (Section 1.2) and future outlooks in conjunction with the developing technology.

1.3.1 Traditional Concept with Real-Time Hardware Correlators

The traditional concept of a GNSS receiver is depicted in Figure 1.4(a). The sampled signal ($\mathbf{y}$) is being processed by a bank of correlators, each of which performing the signal replica generation, Doppler removal, multiplication of the signal with its replica and the accumulation. The accumulated results are stored into a buffer when the code replica generator overflows its period, this time is named as PRN TIC. At time intervals shorter than a code period, named TIC, these accumulated results are sent to a processor closing the feedback of the tracking loops. The flag of PRN TIC, TIC value and NCO values sampled at TIC time are also transferred to the processor. The processor calculates the discriminator functions, loop filtering and sends the control words to the HW correlators for the code and carrier NCOs.

Depending on the performance of the processor, the other processing steps such as bit synchronization, data decoding&demodulation, pseudorange formation, navigation data storage, and PVT estimation may or may not be accomplished therein. Another host processor can be added. The acquisition can be done by the HW correlators, as a serial or hybrid search, or an acquisition unit for parallel search is often added.

The HW correlators run in parallel and are mostly implemented in an ASIC or FPGA. The communication between these parallel digital circuits and the processor must be in real-time, with latency shorter than the time between two successive TICs, which is less than 1 ms for most of the GNSS signals. If an operating system is accommodated in the CPU, it must have real-time capabilities.

Typical commercial GNSS receiver features ASIC correlators and a host processor in a single chip - being a low size, low power consumption and low cost solution. The limitation is in no possible reconfiguration of the ASIC correlators. On the one hand, implementing the correlators into an FPGA will increase the size, power consumption and variable costs, but on the other hand will enable reconfigurability and shorten the development process. Xilinx has recently introduced an architecture, named *Zynq* [91], where a large capacity FPGA is integrated in a single chip with a dual core 800 MHz ARM-based processor - dual Cortex A9. The two CPUs may run in an asymmetric multiprocessing mode where one of the processors uses a convenient OS - e.g. Linux, and the second processor has no OS (bare-metal mode) or a real-time OS to catch-up with the fast data. A big advantage of this architecture is that the FPGA can be reconfigured by the processor.

Except for the conventional architecture, the traditional concept can also incorporate the vector tracking architecture. The challenging task lies more in programming issues where the data from the PVT estimation must be transformed to the control words of the NCOs.

1.3.2 Software-Defined-Radio Concept

The concept of a software-defined-radio (SDR) GNSS receiver is depicted in Figure 1.4(b). The signal samples ($\mathbf{y}$) are stored into a memory at large blocks. The processor then accesses this data as a whole and performs all the processing steps itself. One of the advantage of the SDR concept is its easy reconfiguration and availability of the development tools, specifications, compilers, debuggers, etc. Another advantage is that it can accommodate any of the receiver architectures introduced in Section 1.3, since the same sampled data can be accessed many times unlike in real-time HW correlators.

Even though some of the SDR concepts running on a single CPU in real-time with 1-bit quantization and low sampling frequency appeared in history, most of the today's research receivers rely on a massive parallelism offered by graphic processing units (GPU) [14, 92–96] or on powerful multi-core processors [66]. The authors of GPU-based GNSS receivers [14, 92–96] implemented their algorithms into NVIDIA GPUs using CUDA language for not only graphic purposes. The fact that NVIDIA GPUs are solely integrated in personal computers or in laptops did not allow this concept penetrate the mass market of embedded devices. Nonetheless, several CPU vendors claimed that they are about to integrate GPU on a single chip with the CPU, thus spawning a platform for various advanced embedded applications such as GNSS receivers.

A GPU is composed of a large number of relatively simple arithmetic and logic units (ALUs), local caches and control logic that can operate in parallel, whereas a CPU has a large cache, control logic and a small number of powerful ALUs [97]. The GPUs are suitable for signal processing algorithms where simple arithmetic and logic operations are needed.

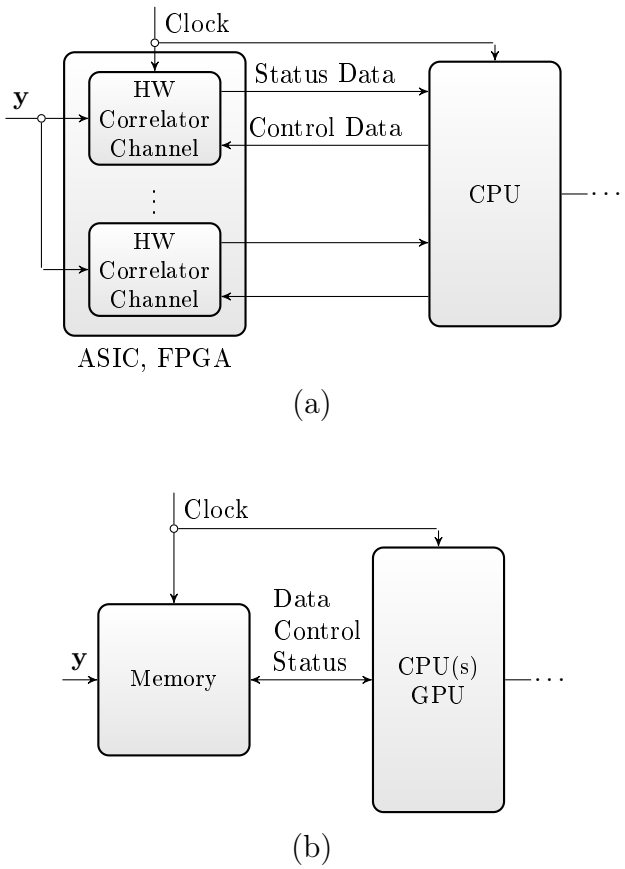


Figure 1.4: Receiver hardware concepts: (a) traditional concept with real-time hardware correlators, (b) software-defined-radio concept

Chapter 2

Testing Platform - the Witch Navigator Receiver

In this section, we discuss the Witch Navigator project. Its development is another crucial part of the study. We will demonstrate the high universality of the receiver and show that it can easily serve as a testing platform for the proposed algorithms in the thesis.

The Witch Navigator can implement both the traditional concept with HW correlators and the SDR concept. Similar university research projects include: the open *Namuru* GNSS receiver [98, 99] based on an Altera FPGA and a host processor being on a small board but with limited performance and reconfigurability, the *ipexSR* receiver [15], the receiver developed by the *PLAN group* at the University of Calgary [95] and others, e.g. [66, 93, 96], based on GPU or multi-core signal processing.

2.1 The Witch Navigator Project

The Witch Navigator (WNav) is an open source project aiming to develop a low-cost high-performance GNSS software receiver capable to process most of the present and future GNSS signals [16, 38–40].

The receiver consists of an *ExpressCard receiver* (Fig. 2.1), hosting two highly reconfigurable RF front-end channels [100] and FPGA-based universal correlators optimized to process majority of the GNSS signals [42], and *PC or notebook*. The high throughput and prompt communication over the PCI Express interface enables that the PC (or notebook) can serve as a control unit for the correlators, accomplish the acquisition, data decoding and PVT estimation. Optionally, the signal snapshots may be continuously transferred to the PC in real time, and fully processed there.

Figure 2.1: The Witch Navigator - ExpressCard photo

The flexible architecture *enables layering ExpressCards*, thus increasing the number of RF channels and number of correlators. Undoubtedly, WNav becomes a multi-system, multi-frequency, and multi-antenna GNSS receiver. The receiver further features *connectors for inertial sensors and external frequency standard*.

In other words, WNav can adopt all the receiver architectures discussed in Section 1.3. For utilization of the SDR concept, the ExpressCard(s) should be connected to a powerful PC with a multi-core CPU or with a GPU suitable for general purpose computing. The receiver is intended for education, research, scientific and small scale applications.

2.2 Open Source Philosophy

The open source philosophy of the project stems from the following facts. All the supporting *source code is freely downloadable and home-reeditable*. The communication driver and application programs are developed in the C language for the Linux operating system with real-time patch. These and all other development tools, including the hardware part, are also free of charge.

2.3 Hardware

The block diagram of a single ExpressCard is depicted in Figure 2.2, PCB layout in Figure 2.3. Each card contains:

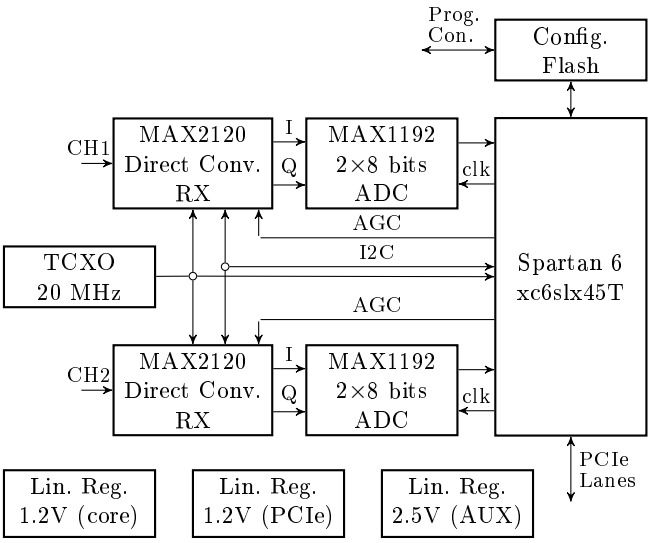


Figure 2.2: The Witch Navigator - ExpressCard block diagram

- two direct conversion receivers
- two dual channel 8-bit ADCs
- Spartan 6 FPGA with PCIe bus
- configuration flash
- highly stable quartz oscillator (20 MHz, 1 ppm)
- linear voltage regulators
- connector for interconnection of the receivers to the large system
- connector for external sensor or device.

The direct conversion receivers (MAX2120), can each be reconfigured from the PC. Their synthesizer's frequency spans over the range of 925 ÷ 2175 MHz with 4 ÷ 40 MHz adjustable baseband filter. The integrated circuits feature voltage controlled amplifier with 75 dB dynamic range, and 20 dB reconfigurable baseband amplifier. Active antenna should be connected to the RF channel inputs (CH1, CH2) via MMCX connector.

The block diagram of the FPGA processor is depicted in Fig. 2.4. The universal correlators (UCorIP) communicate with the PC via the PCI express interface

Figure 2.3: The Witch Navigator - ExpressCard PCB photos

(PCIeIIP, PCIe), clocked by phase lock loop (PLL125). The I2C controller serves as an interface between the direct conversion receivers and the PC, control and status data words are there wrapped into PCIe packets.

The Universal correlator (UCorIP) is an economic implementation of the QPSK E-L correlator. It utilizes RAM-based PRN generators of maximal length of 10230 chips. Each PRN memory can be loaded with data from the PC when in operation. The signals that can be processed are listed in Table 2.1. The currently used FPGA (Spartan 6) has 24 of these E/L correlators. In other words, a single ExpressCard can process up to 24 QPSK signals in parallel using the FPGA. Another part of UCorIP

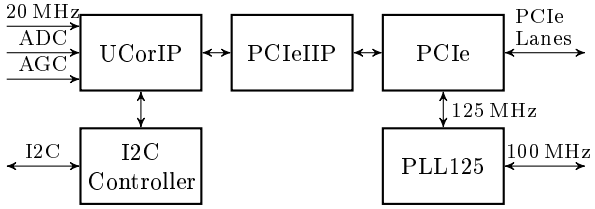


Figure 2.4: The Witch Navigator - FPGA processor

Table 2.1: Universal correlators - possible signals to process

System	**Signal**	**Support.**	**No. of Correl.**	**Note**
GPS	L1 C/A	Yes	1	Verified
	L2C	Partially	1	Data channel only
	L5	Yes	1	
	L1C	Partially	1	BOC(1,1) component only
GLONASS	L1	Yes	1	Verified
	L2	Yes	1	Verified
	L1 K	Not known	1÷2	GLONASS K sat. sig. not specified
	L5 K	Not known	1÷2	GLONASS K sat. sig. not specified
Galileo	E1b & E1c	Yes	1	QPSK method, BOC(6,1) sig. neglected, verified
	E5	Yes	2	Two QPSK sub-carriers [41]
	E5A	Yes	1	Verified
	E5B	Yes	1	Verified
COMPASS	B1	Yes	1	Verified
	B2	Yes	1	Verified

is the signal capture unit which captured samples (=snapshots) are transmitted on the PCIe bus using direct memory access (DMA) at every time interval counter (TIC) event, occurring every 0.8 ms. After the DMA transfer finishes, the PC is interrupted by the FPGA and software handling begins. The situation is depicted in Fig. 2.6. The control words produced by the PC are sent back on the bus at the end of the software handling, which is not illustrated in the figure.

The block diagram of a Universal correlator is depicted in Figure 2.5. The received signal samples y_n are first multiplied by the complex conjugate of the estimated carrier replica and then by the PRN code sequences (Early/Late, Inphase/Quadrature). The results are integrated for the time of one PRN code period (=memory length). The results are dumped at the events of PRN code overflow (PRN TIC Early/Late). The correlator output values are then exported when the TIC increments. At that time, new values of carrier and code control words (b_{carr}, b_{code}) are loaded to the corresponding NCOs and the actual phases of the NCOs are exported as carrier and code measurements, respectively. The late replicas are generated by shifting the early ones as well as the PRN TIC Late is just a shifted version of PRN TIC Early. The distance (Corr. space) can be set at multiples of sampling period ($T_p = 50$ ns). The exported values of a Universal correlator include the real and imaginary part of the integrated values at the last PRN TIC event, flags PRN TIC Early/Late, the value of the TIC, and NCO carrier and code phase measurements. The feed back of the tracking loops is closed via the host PC where discriminators and loop filter are implemented.

2.4 Software

The PC software is primarily designed for the conventional architecture (=UCorIP in use). Transition to the pure software-defined-radio architecture is straightforward (=processing fully by the PC).

The requirements on low latency interrupt handling routing and open source project resulted in selection of the Linux operating system (OS) with real-time (RT) preemptive patch. RT preemptive patch changes a standard Linux kernel into a soft RT kernel. Soft RTOS is an OS that does not guarantee the RT behavior, however, with a high probability will behave like RTOS. The advantages it brings are

- no extra RT application programming interface (API) is needed

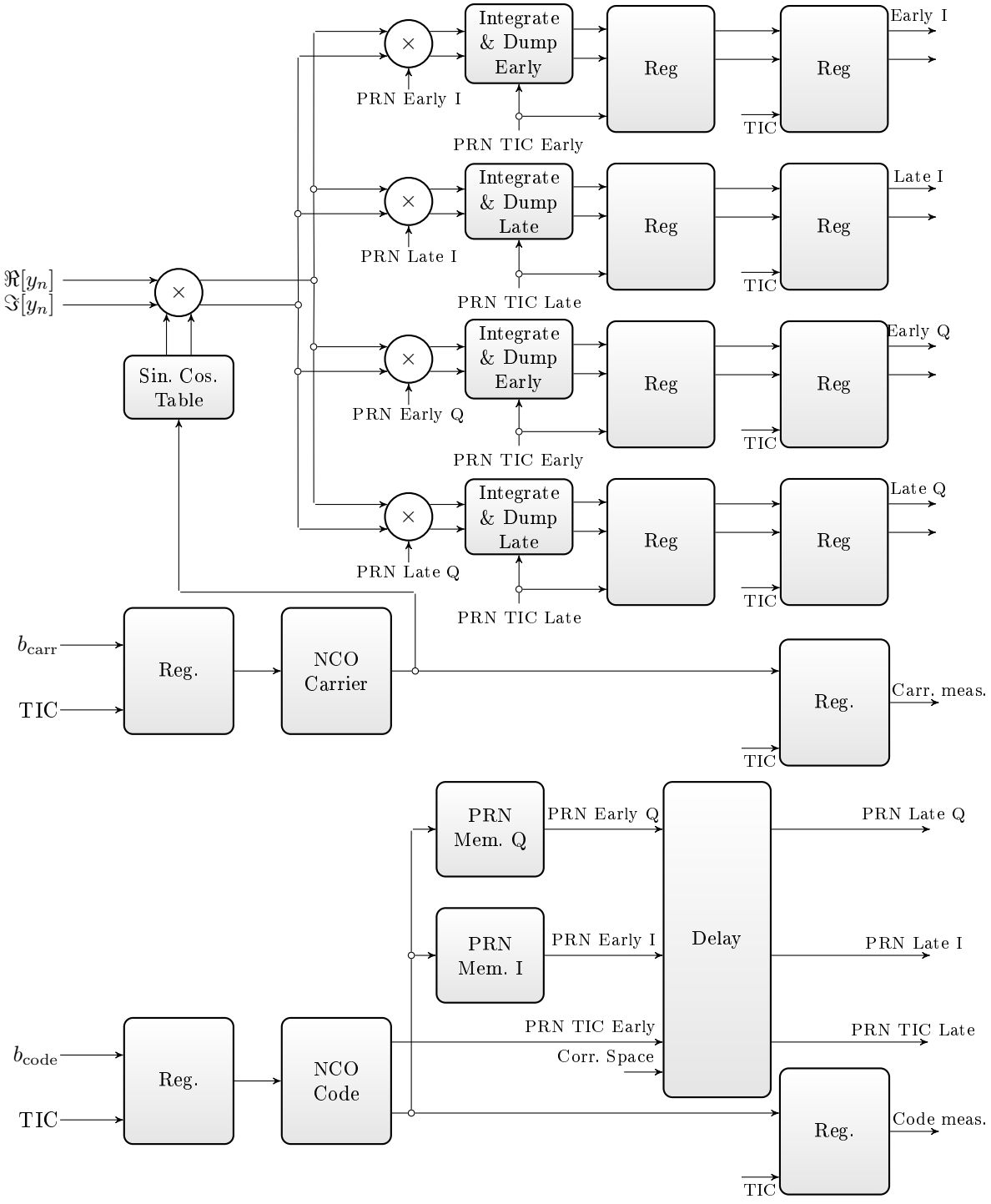


Figure 2.5: Universal Correlator (UCorIP)

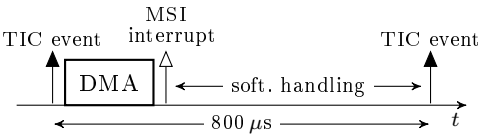


Figure 2.6: The Witch Navigator - time relations of the data transfer

- the user is unconstrained to utilize common standard libraries, compilers, debuggers, etc.

In case of an occasional outage, an error handling routine is implemented.

The PC software diagram is depicted in Fig. 2.7. The developed PCI Express driver operates in the kernel space while other processes, including the FPGA handling process, acquisition process, and PVT process are accessible in the user space. The FPGA handling process is a real-time process with the highest priority. The interrupt handling routine is implemented as a blocking read cycle, which is being unblocked by the interrupt. The interface between the user and the running software is provided via a monitoring process with its graphical user interface.

2.5 Current State

At the moment, the Witch Navigator is able to estimate its PVT using GPS L1 C/A, and GLONASS L1&L2 signals. A single ExpressCard has been used so far and the following combinations are now functional - GPS L1 C/A and GLONASS L1 multi-constellation receiver or GLONASS L1&L2 dual frequency receiver. GPS L2C, Galileo FNAV, INAV modules are under construction, yet the acquisition and tracking modules of Galileo E1, E5, COMPASS B1, B2 signals have been developed. The acquisition supports both serial, and parallel algorithms based on FFT (PCSS, DBZP). The tracking module is implemented using separate DLL/FLL/PLL. The PVT estimation module has implemented the least squares (LS), weighted least squares (WLS), extended Kalman filter (EKF) algorithms, and a novel iterative factor-graph-based (FG) filtering algorithm with Gaussian PDF representation. The receiver supports cold and hot start, warm start is being implemented.

2.6 Future Plans

In the very near future, the project is planned to undergo the following changes:

- cooperation of multiple ExpressCards

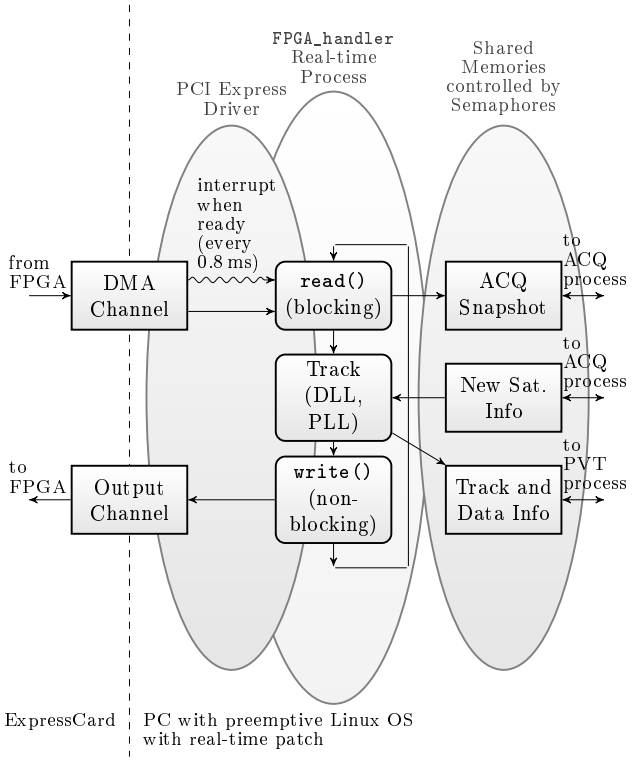


Figure 2.7: The Witch Navigator - PC software diagram

- employment of a higher capacity FPGA
- development of OpenCL libraries for fast acquisition and tracking
- at some user space programs transition to an object-oriented programming language (C++, Python,...)

Long term plans include:

- reception of other GNSS signals, namely GPS L5C, COMPASS signals
- reception of SBAS signals, including EGNOS, WASS, QZSS
- implementation of vector tracking
- implementation of RAIM
- implementation of multipath mitigation techniques, non-line-of-sight positioning techniques
- join of other enthusiastic people, organizations cooperating on the development.

We recall that the main application area of the Witch Navigator lies in education, research, and scientific small scale applications. The project does not aim to compete with commercial companies, but will likely provide materials useful for the development of their projects. The project is not funded by any institution and is politically independent.

2.7 Student's Project Tasks and Responsibilities

The proposing student has the following responsibilities and tasks concerning the Witch Navigator project:

- RF front-end control
- frame synchronization, data decoding & demodulation
- navigation data storage and pseudorange formation
- postprocessing signal acquisition, tracking, bit synchronization
- PVT estimation/filtering

- cooperation on the selection of RTOS and on the interprocess communication (IPC)
- others (web design, templates, logos,...).
- involvement of the university undergraduate students.

Chapter 3

Overview of Estimation Theory

In this chapter, we overview the estimation theory, mostly citing [43] - "the bible of estimation theory for engineers". We address the problem of parameter estimation, discuss desired estimator properties, and basic performance characteristics. Two types of estimators - classical and Bayesian are briefly introduced. For both types of estimators, we discuss the fundamental lower bounds of a minimum square error, so called Cramer-Rao lower bound (CRLB), that the best estimator can attain if it exists. In most cases, other algorithms either suboptimal or with different optimality criterion must be employed which we discuss, as well.

3.1 Problem Definition

The problem of estimation theory is to extract the useful information from noisy continuous time signals. However, discrete time signals are processed by today's digital signal processors and also continuous time signals can be represented with discrete signals using an appropriate orthogonal transformation [101]. Suppose that we have N-point data set arranged in a vector $\mathbf{x} = [x_0\, x_1 \ldots x_{N-1}]^T$ which depends on a unknown, generally vector of $p \times 1$ size, parameter $\boldsymbol{\theta} = [\theta_1 \ldots \theta_p]^T$ which we wish to determine from the data set or define an estimator

$$\hat{\boldsymbol{\theta}} = \mathbf{g}\left(\mathbf{x}\right) \tag{3.1}$$

where $\mathbf{g}$ is some p-dimensional vector function. Given the signal model, its PDF description, we would like to derive somehow an estimator being optimal in some sense, e.g. would minimize the mean square error (MSE)

$$\operatorname{mse}\left[\hat{\boldsymbol{\theta}}\right] = \mathrm{E}\left[\left(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right)^2\right]. \tag{3.2}$$

It is desirable to have estimators that on average converge to the true value - are *unbiased*

$$\mathrm{E}\left[\hat{\boldsymbol{\theta}}\right] = \boldsymbol{\theta}. \tag{3.3}$$

For such estimators, the minimization of their variance also minimizes their MSE, since it can be easily shown that

$$\mathrm{mse}\left[\hat{\boldsymbol{\theta}}\right] = \mathrm{var}\left[\hat{\boldsymbol{\theta}}\right] + b^2\left(\hat{\boldsymbol{\theta}}\right) \tag{3.4}$$

where $b\left(\hat{\boldsymbol{\theta}}\right) = \mathrm{E}\left[\hat{\boldsymbol{\theta}}\right] - \boldsymbol{\theta}$ is the bias of the estimator. If an unbiased estimator cannot be found, we wish to have an estimator that is at least asymptotically unbiased

$$\lim_{N\to\infty}\mathrm{E}\left[\hat{\boldsymbol{\theta}}\right] = \boldsymbol{\theta} \tag{3.5}$$

and its variance goes to zero for a large number of observations

$$\lim_{N\to\infty}\mathrm{var}\left[\hat{\boldsymbol{\theta}}\right] = 0. \tag{3.6}$$

Such estimator is said to be *consistent*.

In *classical estimation theory*, it is assumed that $\boldsymbol{\theta}$ is a deterministic parameter. The complete description of the signal model is then the likelihood PDF $p\left(\mathbf{x}|\boldsymbol{\theta}\right)$. In *Bayesian philosophy*, the estimators have some a priori knowledge about $\boldsymbol{\theta}$ that can improve the performance of the estimator significantly. The complete description of the model is then the joint PDF of the data and the parameter $p\left(\mathbf{x},\, \boldsymbol{\theta}\right) = p\left(\mathbf{x}|\boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right)$. For either class of estimator, the bound of minimum achievable variance of an unbiased estimator can be derived using the *Cramer-Rao lower bound* (CRLB) theorem. An estimator that attains the bound is said to be *efficient*.

If the data come to the estimator gradually and the estimates are to be obtained for every new observation, to lower the dimensionality one should resort to *sequential estimation* where the current estimate is calculated using the previous estimate or its statistical description, and only the new data. In Bayesian philosophy, the parameter often depends on time. If estimates at the current time are obtained using the a priori knowledge about the parameter at that time and the current data, this is referred as *filtering*. If older estimates are recalculated from the new data, this is named as *smoothing*. Extrapolating estimates to the future is referred as *prediction*.

In Bayesian estimators, the use of a priori PDF sometimes allows us to eliminate parameters that we do not want to estimate or cannot estimate. These parameters

are called as *nuisance*. If the vector parameter $\boldsymbol{\theta}$ can be partitioned into a vector of nuisance parameters $\boldsymbol{\theta}_n$ and a vector of parameters of our interest $\boldsymbol{\theta}_s$

$$\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta}_n \\ \boldsymbol{\theta}_s \end{bmatrix} = \begin{bmatrix} r \times 1 \\ (p-r) \times 1 \end{bmatrix} \tag{3.7}$$

the nuisance parameters can be eliminated from the joint PDF as follows

$$p(\mathbf{x}, \boldsymbol{\theta}_s) = \int p(\mathbf{x}, \boldsymbol{\theta}_s, \boldsymbol{\theta}_n) \, \mathrm{d}\boldsymbol{\theta}_n \tag{3.8}$$

$$= \int p(\mathbf{x}, \boldsymbol{\theta}_s | \boldsymbol{\theta}_n) \, p(\boldsymbol{\theta}_n) \, \mathrm{d}\boldsymbol{\theta}_n. \tag{3.9}$$

In signal processing, our signal $\mathbf{s}(\boldsymbol{\theta})$ depending on the true parameter $\boldsymbol{\theta}$ is mostly embedded in additive noise $\mathbf{w}$, denoting as

$$\mathbf{x} = \mathbf{s}(\boldsymbol{\theta}) + \mathbf{w} \tag{3.10}$$

where $\mathbf{s}$ is an N-dimensional deterministic function, $\mathbf{w}$ is an $N \times 1$ random variable with some statistical description. The situation is depicted in Figure 3.1. A tractable and often well representing model is that $\mathbf{w}$ is Gaussian with mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{C}$

$$p(\mathbf{w}) = \mathcal{N}_{\mathbf{w}}(\boldsymbol{\mu}, \mathbf{C}) \tag{3.11}$$

$$= \frac{1}{(2\pi)^{\frac{N}{2}} \det^{\frac{1}{2}}[\mathbf{C}]} \exp\left(-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T \mathbf{C}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right). \tag{3.12}$$

Especially for observations where the covariance matrix does not change over elements, the noise is modeled as zero mean white Gaussian noise (WGN). The covariance matrix is then diagonal with $\sigma^2 > 0$

$$\boldsymbol{\mu} = \mathbf{0} \tag{3.13}$$

$$\mathbf{C} = \mathbf{I}\sigma^2. \tag{3.14}$$

Not always can the explicit formula of performance be derived in a closed form. The most straightforward approach is then to employ Monte Carlo simulations and get the first and second order characteristics of the estimator. It is crucial to use a sufficient number of repetitions to trustfully estimate the characteristics [52]. Along with the performance, we are always interested in complexity of the estimator. Typically, how many floating-point operations (flops) it requires, what is the total memory

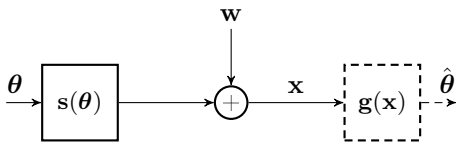


Figure 3.1: Estimator of a signal $\mathbf{s}$ depending on the parameter of our interest $\boldsymbol{\theta}$ embedded in noise $\mathbf{w}$. The estimator takes the input signal $\mathbf{x}$ and gives estimates of the parameter $\hat{\boldsymbol{\theta}}$ through a function $\mathbf{g}$. The signal model is denoted with full lines, whereas the estimator part is denoted dashed.

consumption, etc. Furthermore, we are also interested in how the algorithm is distributed or how it can be parallelized. The other questions are how complicated operations must be adopted and what is the minimal bit width for the operations. These parameters are important when implementation not only in FPGAs, ASICs or GPUs.

In practice, engineers have limited background depending on their focus and cannot implement all existing algorithms, even though they have equipment to do so. The intellectual complexity plays, undoubtedly, an important role when deciding to implement an algorithm.

3.2 Classical Estimators

3.2.1 Minimum Variance Unbiased (MVU) Estimator and Cramer-Rao Lower Bound (CRLB)

The most common requirements we can have on the estimator is that it is unbiased and of minimum variance of the estimate. Such estimators are called as *minimum variance unbiased* (MVU) estimators. The conditions we have are

$$\hat{\boldsymbol{\theta}} = \operatorname{argmin}_{\check{\boldsymbol{\theta}}} \mathrm{E}\left[\left(\check{\boldsymbol{\theta}} - \boldsymbol{\theta}\right)^2\right] \tag{3.15}$$

$$\mathrm{E}\left[\hat{\boldsymbol{\theta}}\right] = \boldsymbol{\theta}. \tag{3.16}$$

However, such estimators do not always exist, since the covariance matrix of the estimate may depend on the true parameter $\boldsymbol{\theta}$ and may yield various estimators for different values of $\boldsymbol{\theta}$. It is not clear which estimator is the best, since we do not know the true value. If the variance is smaller for some $\boldsymbol{\theta}$ and for some $\boldsymbol{\theta}$ larger, the MVU estimator cannot be found, see Fig. 3.2.

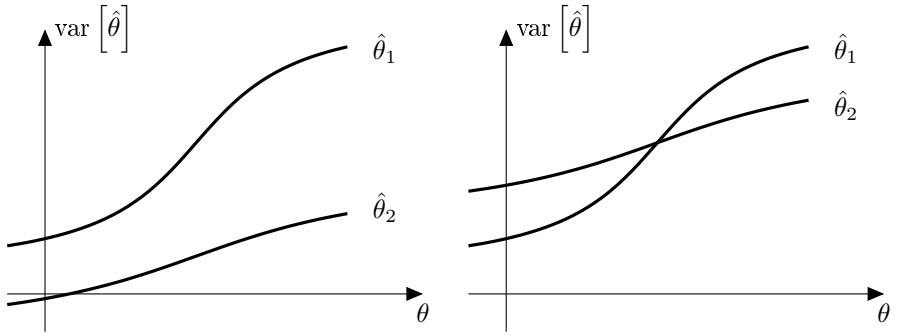


Figure 3.2: Existence of the MVU estimator (a) exists - variance of $\hat{\theta}_1$ is always larger than variance $\hat{\theta}_2$ (b) does not exist - variance of $\hat{\theta}_1$ is for some θ larger than variance $\hat{\theta}_2$ and for some θ lower.

If the estimator exists, we may not be able to find it. Standard procedures to find such an estimator are in order:

- determine the CRLB and check to see if some estimator satisfies it
- apply Rao-Blackwell-Lehmann-Scheffe (RBLS) theorem
- restrict the estimator to be linear (produces the estimator only if the MVU estimator is linear in data).

The *Cramer-Rao lower bound* (CRLB) is a fundamental bound in classical estimation which gives the minimal variance that can have the best estimator. There cannot be any estimator with better performance, except for those using a priori knowledge. If $\hat{\boldsymbol{\theta}}$ is an unbiased estimator of $\boldsymbol{\theta}$, then

$$\mathbf{C}_{\hat{\boldsymbol{\theta}}} \geq \mathbf{J}^{-1}(\boldsymbol{\theta}) \tag{3.17}$$

where $\mathbf{J}(\boldsymbol{\theta})$ is the $p \times p$ Fisher information matrix

$$[\mathbf{J}(\boldsymbol{\theta})]_{ij} = -\mathrm{E}\left[\frac{\partial^2 \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial\theta_i \partial\theta_j}\right]. \tag{3.18}$$

Further if the following regularity condition holds

$$\mathrm{E}\left[\frac{\partial \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right] = \mathbf{0} \tag{3.19}$$

an estimator may be found that attains the CRLB $\mathbf{C}_{\hat{\boldsymbol{\theta}}} = \mathbf{J}^{-1}(\boldsymbol{\theta})$ if and only if

$$\frac{\partial \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{J}(\boldsymbol{\theta})(\mathbf{g}(\mathbf{x}) - \boldsymbol{\theta}) \tag{3.20}$$

and the estimator is $\hat{\boldsymbol{\theta}} = \mathbf{g}(\mathbf{x})$.

If the signal model is linear

$$\mathbf{x} = \mathbf{H}\boldsymbol{\theta} + \mathbf{w} \tag{3.21}$$

where $\mathbf{H}$ is $N \times p$ matrix and $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$, the estimator that attains the CRLB can be found in a closed form

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{C}^{-1}\mathbf{x} \tag{3.22}$$

and the covariance matrix is

$$\mathbf{C}_{\hat{\boldsymbol{\theta}}} = \left(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H}\right)^{-1}. \tag{3.23}$$

Also it holds that

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta},\, \left(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H}\right)^{-1}\right). \tag{3.24}$$

The CRLB for a transformed parameter $\boldsymbol{\alpha} = \mathbf{g}(\boldsymbol{\theta})$ where $\mathbf{g}$ is an r-dimensional function can be found as

$$\mathbf{C}_{\hat{\boldsymbol{\alpha}}} \geq \frac{\partial \mathbf{g}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\mathbf{J}^{-1}(\boldsymbol{\theta})\frac{\partial \mathbf{g}^T(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}. \tag{3.25}$$

In the case of Gaussian observations where the mean and the covariance may depend on $\boldsymbol{\theta}$

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\theta}), \mathbf{C}(\boldsymbol{\theta})) \tag{3.26}$$

the Fisher information matrix can be obtained as

$$\begin{aligned}[\mathbf{J}(\boldsymbol{\theta})]_{ij} &= \left[\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial \theta_i}\right]^T \mathbf{C}^{-1}(\boldsymbol{\theta})\left[\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial \theta_j}\right] \\ &+ \frac{1}{2}\mathrm{tr}\left[\mathbf{C}^{-1}(\boldsymbol{\theta})\frac{\mathbf{C}(\boldsymbol{\theta})}{\partial \theta_i}\mathbf{C}^{-1}(\boldsymbol{\theta})\frac{\mathbf{C}(\boldsymbol{\theta})}{\partial \theta_j}\right].\end{aligned} \tag{3.27}$$

For wide-sense stationary (WSS) processes, the Fisher information matrix can be asymptotically approximated as

$$[\mathbf{J}(\boldsymbol{\theta})]_{ij} \overset{a}{=} \frac{2}{N}\int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{\partial \ln \mathcal{S}_x(f, \boldsymbol{\theta})}{\partial \theta_i}\frac{\partial \ln \mathcal{S}_x(f, \boldsymbol{\theta})}{\partial \theta_j}\mathrm{d}f \tag{3.28}$$

where $\mathcal{S}_x$ is the PSD of zero mean x_n.

For a large number of estimators, it is sufficient to operate with a reduced-dimensionality function $\mathbf{T}(\mathbf{x})$ instead of large data $\mathbf{x}$. Function $\mathbf{T}(\mathbf{x})$ is sufficient, if $p(\mathbf{x}|\mathbf{T}(\mathbf{x}), \boldsymbol{\theta})$ does depend on $\boldsymbol{\theta}$. If it did, we could infer some additional information about $\boldsymbol{\theta}$ from $\mathbf{x}$ compared to the knowledge of $\mathbf{T}(\mathbf{x})$. In this case, $\mathbf{T}(\mathbf{x})$ is called *sufficient statistic* for $\boldsymbol{\theta}$. The *Neyman-Fisher factorization theorem* states that if we can factor the PDF $p(\mathbf{x}|\boldsymbol{\theta})$ as

$$p(\mathbf{x}|\boldsymbol{\theta}) = g(\mathbf{T}(\mathbf{x}), \boldsymbol{\theta}) h(\mathbf{x}) \tag{3.29}$$

where g is a function depending on $\mathbf{x}$ only through $\mathbf{T}(\mathbf{x})$, an $r \times 1$ statistic, and also on $\boldsymbol{\theta}$, and h is a function depending only on $\mathbf{x}$, then $\mathbf{T}(\mathbf{x})$ is a sufficient statistic for $\boldsymbol{\theta}$. Conversely, if $\mathbf{T}(\mathbf{x})$ is a sufficient statistic for $\boldsymbol{\theta}$, then the PDF can be factored as in (3.29).

The *Rao-Blackwell-Lehmann-Scheffe* (RBLS) theorem then gives an approach how to find a MVU estimator based on the sufficient statistic: If $\hat{\boldsymbol{\theta}}$ is an unbiased estimator of $\boldsymbol{\theta}$ and $\mathbf{T}(\mathbf{x})$ is an $r \times 1$ sufficient statistic for $\boldsymbol{\theta}$, then $\hat{\boldsymbol{\theta}} = \mathrm{E}\left[\check{\boldsymbol{\theta}}|\mathbf{T}(\mathbf{x})\right]$ is a valid estimator for $\boldsymbol{\theta}$ not dependent on $\boldsymbol{\theta}$, unbiased, of lesser or equal variance than that of $\check{\boldsymbol{\theta}}$. Additionally, if the sufficient statistic is complete, then $\hat{\boldsymbol{\theta}}$ is the MVU estimator. Completeness means that for a $\mathbf{v}(\mathbf{T})$, an arbitrary $r \times 1$ function of $\mathbf{T}$, if

$$\mathrm{E}[\mathbf{v}(\mathbf{T})] = \int \mathbf{v}(\mathbf{T}) p(\mathbf{T}|\boldsymbol{\theta}) \, \mathrm{d}\mathbf{T} = \mathbf{0} \tag{3.30}$$

for all $\boldsymbol{\theta}$, then it must be true that

$$\mathbf{v}(\mathbf{T}) = \mathbf{0} \tag{3.31}$$

for all $\mathbf{T}$.

3.2.2 Best Linear Unbiased Estimator (BLUE)

If it is impossible to derive the MVU estimator or the evaluation is complex, it is straightforward to find a minimum variance unbiased estimator that is linear in data. We suppose the estimator has the form

$$\hat{\boldsymbol{\theta}} = \mathbf{A}\mathbf{x} \tag{3.32}$$

where $\mathbf{A}$ is a $p \times N$ matrix. The condition that the estimator is unbiased can be expressed as

$$\mathrm{E}\left[\hat{\boldsymbol{\theta}}\right] = \mathbf{A}\mathrm{E}[\mathbf{x}] = \boldsymbol{\theta}. \tag{3.33}$$

In order to satisfy (3.33), the measurement equation must be linear

$$\mathbf{x} = \mathbf{H}\boldsymbol{\theta} + \mathbf{w} \tag{3.34}$$

and $\mathbf{w}$ is zero mean $\mathrm{E}[\mathbf{w}] = \mathbf{0}$, but generally non-Gaussian. The BLUE is identical to the MVU for the linear model with Gaussian noise in (3.22) with the same covariance matrix (3.23).

3.2.3 Maximum Likelihood Estimator (MLE)

Maximum likelihood estimator is a practical solution if the MVU estimator is difficult to find, cannot be found or is complex to implement. It attains the CRLB for large data vectors. The estimator maximizes the likelihood function $p(\mathbf{x}|\boldsymbol{\theta})$

$$\hat{\boldsymbol{\theta}} = \mathrm{argmax}_{\boldsymbol{\theta}}\, p(\mathbf{x}|\boldsymbol{\theta}) \tag{3.35}$$

or equivalently the log-likelihood function $\ln p(\mathbf{x}|\boldsymbol{\theta})$

$$\hat{\boldsymbol{\theta}} = \mathrm{argmax}_{\boldsymbol{\theta}} \ln p(\mathbf{x}|\boldsymbol{\theta}). \tag{3.36}$$

The necessary condition

$$\frac{\partial \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0} \tag{3.37}$$

to find the global maximum must hold. The MLE asymptotically attains the CRLB

$$\hat{\boldsymbol{\theta}} \overset{a}{\sim} \mathcal{N}\left(\boldsymbol{\theta}, \mathbf{J}^{-1}(\boldsymbol{\theta})\right). \tag{3.38}$$

If a MVU estimator exists, MLE will produce it. The MLE is invariant to a transformation. If $\boldsymbol{\alpha} = \mathbf{g}(\boldsymbol{\theta})$ and $\hat{\boldsymbol{\theta}}$ is a MLE of $\boldsymbol{\theta}$, then $\hat{\boldsymbol{\alpha}} = \mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is the MLE of $\boldsymbol{\alpha}$. Since the likelihood function can be evaluated for the observed data, the maximum of it can be found numerically. There are common iterative procedures assuming appropriate initial guess. These include Newton-Raphson method, scoring approach, and the expectation-maximization (EM) algorithm. The *Newton-Raphson method* tries to solve (3.37) iteratively

$$\mathbf{g}(\boldsymbol{\theta}) = \frac{\partial \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tag{3.39}$$

by first order Taylor approximation about previous guess $\boldsymbol{\theta}_i$ from ith iteration

$$\mathbf{g}(\boldsymbol{\theta}) = \mathbf{g}(\boldsymbol{\theta}_i) + \left.\frac{\mathrm{d}\mathbf{g}(\boldsymbol{\theta})}{\mathrm{d}\boldsymbol{\theta}}\right|_{\boldsymbol{\theta}_i} (\boldsymbol{\theta} - \boldsymbol{\theta}_i). \tag{3.40}$$

The new quess is obtained by solving (3.40) with $\mathbf{g}(\boldsymbol{\theta}) = \mathbf{0}$

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \left[\frac{\mathrm{d}\mathbf{g}(\boldsymbol{\theta})}{\mathrm{d}\boldsymbol{\theta}} \bigg|_{\boldsymbol{\theta}_i} \right]^{-1} \mathbf{g}(\boldsymbol{\theta}_i). \tag{3.41}$$

The *scoring method* approximates the second derivative of the likelihood function by the negative of the Fisher information matrix

$$\frac{\partial^2 \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial^2 \boldsymbol{\theta}} \approx -\mathbf{J}(\boldsymbol{\theta}) \tag{3.42}$$

where the equality holds if $\mathbf{x}$ are independent and identically distributed (IID). The fact that the second derivative is substituted by its expected value may increase stability of the iterations. The method is then

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i + \mathbf{J}^{-1}(\boldsymbol{\theta}_i)\, \mathbf{g}(\boldsymbol{\theta}_i). \tag{3.43}$$

The *EM algorithm* ensures at least convergence to a local maximum under some mild conditions. We suppose that the complete data $\mathbf{x}$ can be decomposed into incomplete data sets $\mathbf{y}_1, \ldots, \mathbf{y}_M$ where $M \in \mathbb{N}$ and there is a many-to-one incomplete to complete data transformation $\mathbf{x} = \mathbf{g}(\mathbf{y}_1, \ldots, \mathbf{y}_M) = \mathbf{g}(\mathbf{y})$. Instead of maximizing $\ln p_{\mathbf{x}}(\mathbf{x}|\boldsymbol{\theta})$, the expected value of $\ln p_{\mathbf{y}}(\mathbf{y}|\boldsymbol{\theta})$ is maximized as a conditional expectation

$$\mathrm{E}_{\mathbf{y}|\mathbf{x}}\left[\ln p_{\mathbf{y}}(\mathbf{y}|\boldsymbol{\theta})\right] = \int \ln p_{\mathbf{y}}(\mathbf{y}|\boldsymbol{\theta}) p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})\, \mathrm{d}\mathbf{y}. \tag{3.44}$$

Instead of using $\boldsymbol{\theta}$ that we do not know, we use ith guess $\boldsymbol{\theta}_i$. We then have the following iterative algorithm:

Expectation:

$$U(\boldsymbol{\theta}|\boldsymbol{\theta}_i) = \int \ln p_{\mathbf{y}}(\mathbf{y}|\boldsymbol{\theta}) p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_i)\, \mathrm{d}\mathbf{y}. \tag{3.45}$$

Maximization:

$$\boldsymbol{\theta}_{i+1} = \mathrm{argmax}_{\boldsymbol{\theta}} U(\boldsymbol{\theta}|\boldsymbol{\theta}_i). \tag{3.46}$$

It is sometimes the case that the data are Gaussian $p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \mathbf{C})$ and large, and maximization of the PDF would be computationally demanding due to the inverse of large $\mathbf{C}$. If $\mathbf{x}$ is a wide-sense stationary (WSS) process, the covariance matrix $\mathbf{C}$ is Toeplitz, the derivative of the log-likelihood can be asymptotically approximated as

$$\frac{\partial \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial \theta_i} \overset{a}{=} -\frac{N}{2} \int_{-\frac{1}{2}}^{\frac{1}{2}} \left[\frac{1}{\mathcal{S}_x(f)} - \frac{\mathcal{P}_x(f)}{\mathcal{S}_x^2(f)} \right] \frac{\partial \mathcal{P}_x(f)}{\partial \theta_i} \mathrm{d}f \tag{3.47}$$

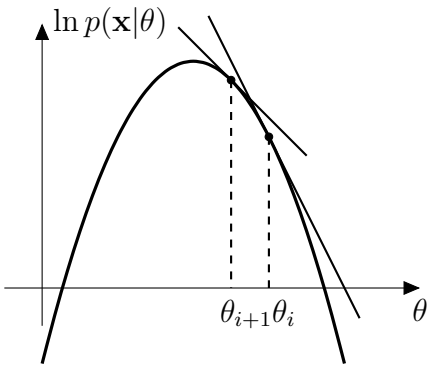


Figure 3.3: Newton-Raphson iterations for scalar parameter θ

where $\mathcal{P}_x(f)$ is the periodogram of the data

$$\mathcal{P}_x(f) = \frac{1}{N}\left|\int_{-\frac{1}{2}}^{\frac{1}{2}} x_n \exp\left(-\mathrm{j}2\pi f n\right) \mathrm{d}f\right|^2 \tag{3.48}$$

and $\mathcal{S}_x(f)$ is the PSD of the data.

3.2.4 Least Squares (LS)

The approach of least squares (LS) minimizes the least squares between the data and the useful signal. No statistical knowledge about the data is needed, only the signal model. However, the statistical performance cannot be assessed. The LS method is widely used since it is easy to implement. Assume the following signal model

$$\mathbf{x} = \mathbf{s}\left(\boldsymbol{\theta}\right) + \mathbf{w} \tag{3.49}$$

where $\mathbf{s}$ is purely deterministic signal depending on $\boldsymbol{\theta}$ and $\mathbf{w}$ is a random variable. The LS estimator (LSE) minimizes the sum of the squared difference between the measured and the deterministic signal

$$\hat{\boldsymbol{\theta}} \;=\; \mathrm{argmin}_{\boldsymbol{\theta}} \left(\mathbf{x} - \mathbf{s}\left(\boldsymbol{\theta}\right)\right)^T \left(\mathbf{x} - \mathbf{s}\left(\boldsymbol{\theta}\right)\right).$$

The approach is reasonable when the noise is zero mean $\mathrm{E}\left[\mathbf{w}\right] = \mathbf{0}$. If the signal model is linear, there is a linear dependence of $\mathbf{s}$ on $\boldsymbol{\theta}$ via an $N \times p$ matrix $\mathbf{H}$

$$\mathbf{s} = \mathbf{H}\boldsymbol{\theta}. \tag{3.50}$$

It is said to generate a *linear least squares* problem that can be solved in a closed form

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{x}. \tag{3.51}$$

Conversely, a *nonlinear least squares problem* is obtained which is mostly solved iteratively or via a grid search. If the signal is linear in some of the parameters of vector $\boldsymbol{\theta}$ and in some nonlinear, a *separate least squares problem* is generated. Weighting can be applied to the squares that are minimized so that we minimize

$$\mathcal{LS}\left(\boldsymbol{\theta}\right) = \left(\mathbf{x} - \mathbf{s}\left(\boldsymbol{\theta}\right)\right)^T\mathbf{W}\left(\mathbf{x} - \mathbf{s}\left(\boldsymbol{\theta}\right)\right) \tag{3.52}$$

where $\mathbf{W}$ is a diagonal $N \times N$ weighting matrix $w_{ii} > 0$. The linear least squares problem is then solved as

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{H}^T\mathbf{W}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{W}\mathbf{x}. \tag{3.53}$$

If the covariance of a zero mean noise $\mathbf{w}$ is known, commonly $\mathbf{W} = \mathbf{C}^{-1}$. The minimum least squares error is then

$$\mathcal{LS}\left(\hat{\boldsymbol{\theta}}\right) = \mathbf{x}\left(\mathbf{W} - \mathbf{W}\mathbf{H}\left(\mathbf{H}^T\mathbf{W}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{W}\right)\mathbf{x}. \tag{3.54}$$

The least squares can be solved recursively in two ways. If even the signal model is unknown, it can be approximated by a polynomial which number of coefficients is increased and estimated with new observed data [43], known as *order-recursive least squares*. Provided the signal model is known, the new estimate can be obtained recursively for the incoming data. The problem is named as *sequential least squares*. It can be used only for uncolored noise, or $\mathbf{C}$ must be diagonal. Assuming that

$$\mathbf{C}_n = \operatorname{diag}\left(\sigma_0^2, \ldots, \sigma_n^2\right) \tag{3.55}$$

$$\mathbf{H}_n = \begin{bmatrix} \mathbf{H}_{n-1} \\ \mathbf{h}_n^T \end{bmatrix} = \begin{bmatrix} n \times p \\ 1 \times p \end{bmatrix} \tag{3.56}$$

$$\mathbf{x}_n = \left[x_0 \ldots x_n\right]^T \tag{3.57}$$

yields a recursive estimator in the following form

$$\hat{\boldsymbol{\theta}}_n = \hat{\boldsymbol{\theta}}_{n-1} + \mathbf{K}_n\left(x_n - \mathbf{h}_n^T\hat{\boldsymbol{\theta}}_{n-1}\right) \tag{3.58}$$

$$\mathbf{K}_n = \frac{\boldsymbol{\chi}_{n-1}\mathbf{h}_n}{\sigma_n^2 + \mathbf{h}_n^T\boldsymbol{\chi}_{n-1}\mathbf{h}_n} \tag{3.59}$$

$$\boldsymbol{\chi}_n = \left(\mathbf{I} - \mathbf{K}_n\mathbf{h}_n^T\right)\boldsymbol{\chi}_{n-1} \tag{3.60}$$

where $\boldsymbol{\chi}_n = \mathbf{C}\left(\hat{\boldsymbol{\theta}}_n\right)$ is the covariance matrix of the estimate at time n. Note that no matrix inversion is needed. To initialize the recursion, $\boldsymbol{\chi}_{-1}$ may be chosen large and $\hat{\boldsymbol{\theta}}_{-1}$ of zeros.

If the parameter is constrained by a linear equation

$$\mathbf{A}\boldsymbol{\theta} = \mathbf{b} \tag{3.61}$$

where $\mathbf{A}$ is an $r \times p$ matrix $\mathbf{b}$ is an $r \times 1$ vector, it can be regarded at least squares criterion to minimize the following Lagrangian

$$\mathcal{LS}\left(\hat{\boldsymbol{\theta}}\right) = \left(\mathbf{x} - \mathbf{H}\boldsymbol{\theta}\right)^T \left(\mathbf{x} - \mathbf{H}\boldsymbol{\theta}\right) + \boldsymbol{\lambda}^T \left(\mathbf{A}\boldsymbol{\theta} - \mathbf{b}\right) \tag{3.62}$$

where $\boldsymbol{\lambda}$ is an $r \times 1$ vector. The constrained estimate $\hat{\boldsymbol{\theta}}_c$ is then

$$\hat{\boldsymbol{\theta}}_c = \hat{\boldsymbol{\theta}} - \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{A}^T\left(\mathbf{A}\left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{A}^T\right)^{-1}\left(\mathbf{A}\hat{\boldsymbol{\theta}} - \mathbf{b}\right) \tag{3.63}$$

where $\hat{\boldsymbol{\theta}}$ is obtained from (3.51).

The nonlinear least squares problem can be solved if the following transformation from nonlinear to linear signal model exists

$$\begin{aligned} \boldsymbol{\alpha} &= \mathbf{g}\left(\boldsymbol{\theta}\right) && (3.64)\\ \mathbf{s}\left(\boldsymbol{\theta}\left(\boldsymbol{\alpha}\right)\right) &= \mathbf{s}\left(\mathbf{g}^{-1}\left(\boldsymbol{\alpha}\right)\right) = \mathbf{H}\boldsymbol{\alpha} && (3.65) \end{aligned}$$

where $\boldsymbol{\alpha}$ is a $p \times 1$ vector and $\mathbf{g}$ is a p-dimensional function with inverse $\mathbf{g}^{-1}$ which must exist. We can find the estimate in two steps

$$\begin{aligned} \hat{\boldsymbol{\alpha}} &= \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{x} && (3.66)\\ \hat{\boldsymbol{\theta}} &= \mathbf{g}^{-1}\left(\hat{\boldsymbol{\alpha}}\right). && (3.67) \end{aligned}$$

The second case of the nonlinear least squares problems discussed is when the parameters can be separated into linear $\boldsymbol{\alpha}$ and nonlinear $\boldsymbol{\beta}$ so that

$$\mathbf{s} = \mathbf{H}\left(\boldsymbol{\alpha}\right)\boldsymbol{\beta} \tag{3.68}$$

where

$$\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} (p-q)\times 1 \\ q \times 1 \end{bmatrix} \tag{3.69}$$

and $\mathbf{H}$ is an $N \times q$ dependent matrix on $\boldsymbol{\alpha}$. The problem then reduces to

$$\begin{aligned} \hat{\boldsymbol{\alpha}} &= \text{argmax}_{\boldsymbol{\alpha}}\, \mathbf{x}^T\mathbf{H}\left(\boldsymbol{\alpha}\right)\left(\mathbf{H}^T\left(\boldsymbol{\alpha}\right)\mathbf{H}\left(\boldsymbol{\alpha}\right)\right)^{-1}\mathbf{H}^T\left(\boldsymbol{\alpha}\right)\mathbf{x} && (3.70)\\ \hat{\boldsymbol{\beta}} &= \left(\mathbf{H}^T\left(\hat{\boldsymbol{\alpha}}\right)\mathbf{H}\left(\hat{\boldsymbol{\alpha}}\right)\right)^{-1}\mathbf{H}^T\left(\hat{\boldsymbol{\alpha}}\right)\mathbf{x}. && (3.71) \end{aligned}$$

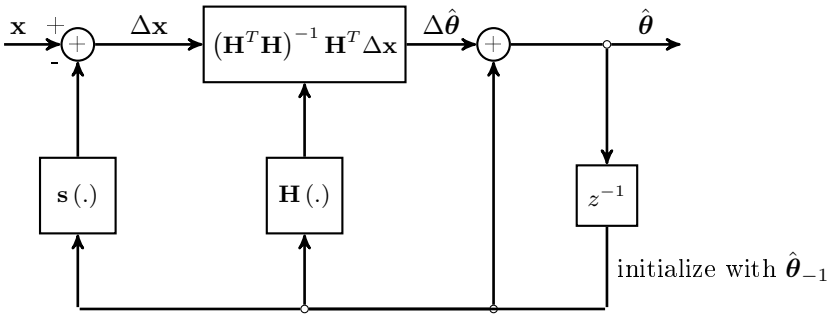


Figure 3.4: Gauss-Newton iterative solution to nonlinear least squares problem.

There are two common iterative methods for nonlinear least squares. The *Newton-Raphson method* iteratively solves the equation of the derivative of the least square criterion that should be zero as a necessary condition for an extreme. It results in the following iterations, with index k denoting the number of the iteration,

$$\begin{aligned}\hat{\boldsymbol{\theta}}_k &= \hat{\boldsymbol{\theta}}_{k-1} + \left(\mathbf{H}^T\left(\hat{\boldsymbol{\theta}}_k\right)\mathbf{H}\left(\hat{\boldsymbol{\theta}}_k\right) - \sum_{n=0}^{N-1}\mathbf{G}_n\left(\hat{\boldsymbol{\theta}}_k\right)\left(x_n - \left[\mathbf{s}\left(\hat{\boldsymbol{\theta}}_k\right)\right]_n\right)\right)^{-1} \\ &\cdot \quad \mathbf{H}^T\left(\hat{\boldsymbol{\theta}}_k\right)\left(\mathbf{x} - \mathbf{s}\left(\hat{\boldsymbol{\theta}}_k\right)\right) \end{aligned} \tag{3.72}$$

where

$$\left[\mathbf{H}\left(\boldsymbol{\theta}\right)\right]_{ij} = \frac{\partial s_i}{\partial \theta_j} \tag{3.73}$$

$$\left[\mathbf{G}_n\left(\boldsymbol{\theta}\right)\right]_{ij} = \frac{\partial^2 s_n}{\partial \theta_i \theta_j}. \tag{3.74}$$

The *Gauss-Newton method* linearizes $\mathbf{s}(\boldsymbol{\theta})$ about the previous estimate $\hat{\boldsymbol{\theta}}_{k-1}$

$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_{k-1} + \left(\mathbf{H}^T\left(\hat{\boldsymbol{\theta}}_k\right)\mathbf{H}\left(\hat{\boldsymbol{\theta}}_k\right)\right)^{-1}\mathbf{H}^T\left(\hat{\boldsymbol{\theta}}_k\right)\left(\mathbf{x} - \mathbf{s}\left(\hat{\boldsymbol{\theta}}_k\right)\right) \tag{3.75}$$

where $\mathbf{H}\left(\boldsymbol{\theta}\right)$ is obtained as in (3.73). The algorithm is depicted in Figure 3.4. Both iterative methods may face similiar convergence problems. They can either converge to any other extreme than a global minimum or may converge slowly or even not at all if the previous estimate is not in a close vicinity of the true value. Therefore, an initial estimate must be selected with care.

3.2.5 Method of Moments

The method of moments produces an estimator that is easy to implement and has good results for long data vectors because the estimator is usually consistent. But the estimator has no optimality properties, however, mostly it can be approximated using a Taylor series expansion [43]. The method of moments can be used to initialize iterative estimators. To obtain the method, let us define kth moment of the observed signal x_n, that is a WSS process,

$$\mu_k = \mathrm{E}\left[(x_n)^k\right] \tag{3.76}$$

and a function h_k that calculates the kth moment from the parameter

$$\mu_k = h_k(\boldsymbol{\theta}). \tag{3.77}$$

Defining vectors, for K as the maximum number of considered moments,

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_K \end{bmatrix} \triangleq \mathbf{h}(\boldsymbol{\theta}) \triangleq \begin{bmatrix} h_1(\boldsymbol{\theta}) \\ \vdots \\ h_K(\boldsymbol{\theta}) \end{bmatrix} \tag{3.78}$$

and assuming that $\mathbf{h}^{-1}$ exists, we can summarize the method of moments as

$$\hat{\boldsymbol{\theta}} = \mathbf{h}^{-1}(\hat{\boldsymbol{\mu}}) \tag{3.79}$$

where

$$\hat{\boldsymbol{\mu}} = \begin{bmatrix} \frac{1}{N}\sum_{n=0}^{N-1} x_n \\ \vdots \\ \frac{1}{N}\sum_{n=0}^{N-1} (x_n)^K \end{bmatrix}. \tag{3.80}$$

3.3 Bayesian Estimators - Minimum Mean Square Error (MMSE) Estimator and Maximum a Posteriori (MAP) Estimator

We assume that $\boldsymbol{\theta}$ is a random variable which particular realization we have to estimate. We can incorporate the a priori knowledge to the estimation to improve the performance. If the MVU estimator cannot be found, we can assign prior PDF to $\boldsymbol{\theta}$ to find an estimator that minimizes the MSE. Such estimator is then said to

be optimal "on the average" and always exists. The MSE in classical estimation is defined as

$$\text{mse}\left[\hat{\boldsymbol{\theta}}\right] = \int \left(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right)^2 p\left(\mathbf{x}|\boldsymbol{\theta}\right) \mathrm{d}\mathbf{x} \tag{3.81}$$

whereas in Bayesian philosophy as

$$\begin{aligned} \text{Bmse}\left[\hat{\boldsymbol{\theta}}\right] &= \int\int \left(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right)^2 p\left(\mathbf{x},\, \boldsymbol{\theta}\right) \mathrm{d}\mathbf{x}\mathrm{d}\boldsymbol{\theta} & (3.82)\\ &= \mathrm{E}_{\boldsymbol{\theta}}\left[\text{mse}\left[\hat{\boldsymbol{\theta}}\right]\right] & (3.83) \end{aligned}$$

Note that $\text{mse}\left[\hat{\boldsymbol{\theta}}\right]$ can generally depend on $\boldsymbol{\theta}$, whereas $\text{Bmse}\left[\hat{\boldsymbol{\theta}}\right]$ cannot, since it is "averaged over $\boldsymbol{\theta}$". The Bayesian MSE estimate can be derived to be the ensemble of the parameter conditioned on the data [43]

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \mathrm{E}\left[\boldsymbol{\theta}|\mathbf{x}\right] & (3.84)\\ &= \int \boldsymbol{\theta} p\left(\boldsymbol{\theta}|\mathbf{x}\right) \mathrm{d}\boldsymbol{\theta}. & (3.85) \end{aligned}$$

The estimator is named as the *minimum mean square error* (MMSE) estimator. The posterior PDF can be obtained from the likelihood PDF $p\left(\mathbf{x}|\boldsymbol{\theta}\right)$ and the prior PDF $p\left(\boldsymbol{\theta}\right)$ applying the Bayesian rule

$$p\left(\boldsymbol{\theta}|\mathbf{x}\right) = \frac{p\left(\mathbf{x}|\boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right)}{\int p\left(\mathbf{x}|\boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}\right) \mathrm{d}\boldsymbol{\theta}} \tag{3.86}$$

with the denominator being just a normalizing factor ensuring proper scaling of the posterior PDF.

If the signal model is linear with Gaussian PDF, the estimator can be found in a closed form. Assume the linear model as in (3.21)

$$\mathbf{x} = \mathbf{H}\boldsymbol{\theta} + \mathbf{w} \tag{3.87}$$

where $\boldsymbol{\theta}$ is now a random parameter with prior Gaussian PDF $\mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}},\, \mathbf{C}_{\boldsymbol{\theta}}\right)$ and $\mathbf{w}$ is a noise vector independent of $\boldsymbol{\theta}$ with PDF $\mathcal{N}\left(\mathbf{0},\, \mathbf{C}_{\mathbf{w}}\right)$. Then the posterior PDF is also Gaussian with mean

$$\mathrm{E}\left[\boldsymbol{\theta}|\mathbf{x}\right] = \boldsymbol{\mu}_{\boldsymbol{\theta}} + \mathbf{C}_{\boldsymbol{\theta}}\mathbf{H}^T \left(\mathbf{H}\mathbf{C}_{\boldsymbol{\theta}}\mathbf{H}^T + \mathbf{C}_{\mathbf{w}}\right)^{-1} \left(\mathbf{x} - \mathbf{H}\boldsymbol{\mu}_{\boldsymbol{\theta}}\right) \tag{3.88}$$

and covariance

$$\mathbf{C}_{\boldsymbol{\theta}|\mathbf{x}} = \mathbf{C}_{\boldsymbol{\theta}} - \mathbf{C}_{\boldsymbol{\theta}}\mathbf{H}^T \left(\mathbf{H}\mathbf{C}_{\boldsymbol{\theta}}\mathbf{H}^T + \mathbf{C}_{\mathbf{w}}\right)^{-1} \mathbf{H}\mathbf{C}_{\boldsymbol{\theta}} \tag{3.89}$$

where $\mathbf{H}$ need to be full rank.

Generally, there is no need to restrict ourselves to a class of Bayesian estimators where just the MSE is minimized. Denoting $\boldsymbol{\epsilon}$ the estimate error $\boldsymbol{\epsilon} = \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}$, we could also minimize $\mathrm{E}\left[\|\boldsymbol{\epsilon}\|\right]$ instead of $\mathrm{E}\left[\|\boldsymbol{\epsilon}\|^2\right]$ which would lead to a different estimator. Defining $\mathcal{C}(\boldsymbol{\epsilon})$ as a *cost function*, a general Bayesian estimator is said to minimize *Bayesian risk* $\mathcal{BR}$

$$\mathcal{BR} = E\left[\mathcal{C}(\boldsymbol{\epsilon})\right]. \tag{3.90}$$

If $\mathcal{C}(\boldsymbol{\epsilon}) = \|\boldsymbol{\epsilon}\|^2$, we get the MMSE estimator. The cost function $\mathcal{C}(\boldsymbol{\epsilon}) = \|\boldsymbol{\epsilon}\|$ penalizes all errors equally. Another example "hit-or-miss" cost function can be defined as

$$\mathcal{C}(\boldsymbol{\epsilon}) = \begin{cases} 0 & \|\boldsymbol{\epsilon}\| > \delta \\ 1 & \|\boldsymbol{\epsilon}\| \leq \delta \end{cases} \tag{3.91}$$

where $\delta > 0$. These cost functions are depicted in Figure 3.5 for a scalar parameter θ. It can be shown that the Bayesian risk for the absolute value cost function is minimized for the median of the scalar parameter θ conditioned on the data $\mathbf{x}$

$$\int_{-\infty}^{\hat{\theta}} p(\theta|\mathbf{x})\,\mathrm{d}\theta = \int_{\hat{\theta}}^{\infty} p(\theta|\mathbf{x})\,\mathrm{d}\theta \tag{3.92}$$

whereas the "hit-or-miss" minimizes the Bayesian risk for maximum of the posterior PDF, considering scalar parameter θ,

$$\begin{aligned} \hat{\theta} &= \mathrm{argmax}_\theta\, p(\theta|\mathbf{x}) && (3.93)\\ &= \mathrm{argmax}_\theta\, \frac{p(\mathbf{x}|\theta)\,p(\theta)}{p(\mathbf{x})} && (3.94)\\ &= \mathrm{argmax}_\theta\, p(\mathbf{x}|\theta)\,p(\theta). && (3.95) \end{aligned}$$

This estimator is named as *maximum a posteriori* (MAP) estimator. To summarize this point, we recall that the posterior PDF $p(\boldsymbol{\theta}|\mathbf{x})$ is a crucial statistics for deriving Bayesian estimators. For the square cost function, the case of MMSE estimator, the *mean* of the cost function is calculated. In the case of the absolute value cost function, the *median* is needed, and for the "hit-or-miss" cost function, the case of MAP estimator, the *mode* must be found.

Sometimes, we are interested in a MMSE estimate of scalar parameter, say θ_i for $0 < i \leq p$, having the vector parameter $\boldsymbol{\theta}$. There are two equivalent approaches to get the scalar estimate. The first one is to take the scalar parameter estimate from

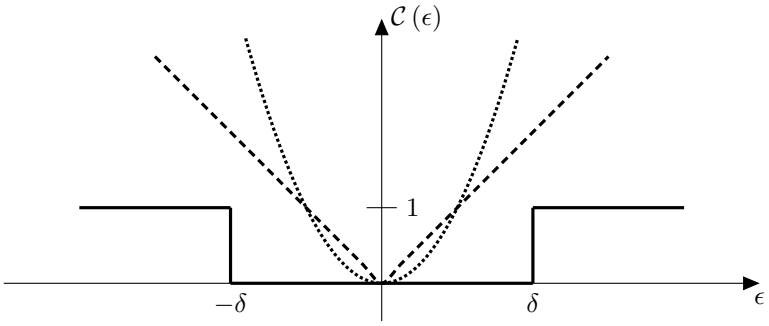


Figure 3.5: Exemplary cost functions for a scalar parameter θ. Dotted line denotes the quadratic cost function $\boldsymbol{\epsilon}^2$, the dashed line denotes the norm cost function $\|\boldsymbol{\epsilon}\|$, the full line denotes the cost function defined in (3.91).

the corresponding element of the vector parameter estimate so that

$$\hat{\theta}_i = \int \theta_i p\left(\boldsymbol{\theta}|\mathbf{x}\right) \mathrm{d}\boldsymbol{\theta} \tag{3.96}$$

$$= \int \ldots \int \theta_i p\left(\boldsymbol{\theta}|\mathbf{x}\right) \mathrm{d}\theta_1 \ldots \mathrm{d}\theta_p. \tag{3.97}$$

But more efficient might be to *marginalize* the desired parameter by eliminating the other parameters assumed as nuisance

$$\hat{\theta}_i = \int \theta_i p\left(\theta_i|\mathbf{x}\right) \mathrm{d}\theta_i \tag{3.98}$$

$$p\left(\theta_i|\mathbf{x}\right) = \int \ldots \int p\left(\boldsymbol{\theta}|\mathbf{x}\right) \underbrace{\mathrm{d}\theta_1 \ldots \mathrm{d}\theta_p}_{\text{except for } \mathrm{d}\theta_i}. \tag{3.99}$$

The MMSE estimator is invariant to linear transformation. If $\boldsymbol{\alpha} = \mathbf{A}\boldsymbol{\theta} + \mathbf{b}$ where $\mathbf{A}$ is an $r \times p$ matrix and $\mathbf{b}$ is an $r \times 1$ vector, then

$$\hat{\boldsymbol{\alpha}} = \mathbf{A}\hat{\boldsymbol{\theta}} + \mathbf{b}. \tag{3.100}$$

The MAP estimate of a vector parameter $\boldsymbol{\theta}$ can calculated as the MAP estimate of all marginalized posterior PDFs of θ_i as in (3.98), (3.99)

$$\hat{\theta}_i = \mathrm{argmax}_\theta \, p\left(\theta_i|\mathbf{x}\right). \tag{3.101}$$

It should be noted that maximizing the posterior PDF of the vector parameter $\boldsymbol{\theta}$ does not generally produce the same estimator as the MAP estimator. However, a

vector MAP estimator can be proposed, and it can be shown that it minimizes a different Bayesian risk,

$$\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}}\, p\left(\boldsymbol{\theta}|\mathbf{x}\right). \tag{3.102}$$

If $\boldsymbol{\theta}$ and $\mathbf{x}$ are jointly Gaussian, then the mean and the peak of the posterior PDF merge and both MAP and MMSE estimators are identical. The covariance matrix of a Bayesian estimator must be regarded to the "averaged" parameter and can be derived to be

$$\mathbf{M}_{\hat{\boldsymbol{\theta}}} = \mathrm{E}_{\boldsymbol{\theta}}\left[\mathbf{C}_{\boldsymbol{\theta}|\mathbf{x}}\right]. \tag{3.103}$$

If $\boldsymbol{\theta}$ and $\mathbf{x}$ are jointly Gaussian, the estimate and its covariance matrix are

$$\hat{\boldsymbol{\theta}} = \mathrm{E}\left[\boldsymbol{\theta}\right] + \mathbf{C}_{\boldsymbol{\theta}\mathbf{x}}\mathbf{C}_{\mathbf{x}\mathbf{x}}^{-1}\left(\mathbf{x} - \mathrm{E}\left[\mathbf{x}\right]\right) \tag{3.104}$$

$$\mathbf{M}_{\hat{\boldsymbol{\theta}}} = \mathbf{C}_{\boldsymbol{\theta}\boldsymbol{\theta}} - \mathbf{C}_{\boldsymbol{\theta}\mathbf{x}}\mathbf{C}_{\mathbf{x}\mathbf{x}}^{-1}\mathbf{C}_{\mathbf{x}\boldsymbol{\theta}}. \tag{3.105}$$

3.3.1 Linear MMSE Estimator and Wiener Filter

Similar to BLUE estimators, we might restrict the estimator to be linear and of the MMSE (LMMSE) for that case. The motivation for this approach is again lower complexity of the estimator. Clearly, the estimator will be suboptimal, except for cases leading to a linear estimator, and will depend only on the mean and the covariance matrix. No PDF is needed to derive the estimator. The LMMSE estimate and its covariance matrix is identical to the MMSE estimate for the jointly Gaussian case as in (3.104), (3.105). *Bayesian Gauss-Markov theorem* states that if the signal model is linear as in (3.87) with general PDFs having the same mean and the covariance matrix as the Gaussian case there, the optimal linear estimator is given by (3.104) with (3.105) covariance matrix.

A sequential LMMSE estimator can be derived, named as *Wiener filter*. Assuming the Bayesian linear model of the data, diagonal covariance matrix $\mathbf{C}_{\mathbf{w}} = \text{diag}\left(\sigma_0^2, \ldots, \sigma_{N-1}^2\right)$, and the following notation for time index $n < N$

$$\mathbf{M}_n = \mathrm{E}\left[\left(\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}_n\right)^T\left(\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}_n\right)\right] \tag{3.106}$$

$$\mathbf{H}_n = \begin{bmatrix} \mathbf{H}_{n-1} \\ \mathbf{h}_n^T \end{bmatrix} = \begin{bmatrix} n \times p \\ 1 \times p \end{bmatrix} \tag{3.107}$$

the recursion to obtain the estimate and its MSE matrix can be summarized as follows:

Estimator Update:

$$\hat{\boldsymbol{\theta}}_n = \hat{\boldsymbol{\theta}}_{n-1} + \mathbf{K}_n \left(x_n - \mathbf{h}_n^T \hat{\boldsymbol{\theta}}_{n-1} \right) \tag{3.108}$$

where

$$\mathbf{K}_n = \frac{\mathbf{M}_{n-1}\mathbf{h}_n}{\sigma_n^2 + \mathbf{h}_n^T \mathbf{M}_{n-1}\mathbf{h}_n}. \tag{3.109}$$

Minimum MSE Matrix Update:

$$\mathbf{M}_n = \left(\mathbf{I} - \mathbf{K}_n \mathbf{h}_n^T \right) \mathbf{M}_{n-1}. \tag{3.110}$$

The common way to initialize the estimator is to set $\hat{\boldsymbol{\theta}}_{-1} = \mathrm{E}\left[\boldsymbol{\theta}\right]$, $\mathbf{M}_{-1} = \mathbf{C}_{\boldsymbol{\theta}\boldsymbol{\theta}}$. Note that no matrix inversions are needed. Interesting results arise for the case where $\mathbf{x} = \boldsymbol{\theta} + \mathbf{w}$ where both $\boldsymbol{\theta}$ and $\mathbf{w}$ are zero mean. It can be shown that the filter becomes time invariant for large n and its impulse response can be found based on the autocorrelation properties of $\boldsymbol{\theta}$ and $\mathbf{w}$. An interested reader should consult [43].

3.3.2 Kalman Filter (KF)

The Kalman filter can be said to generalize the Wiener filters. They apply to nonstationary signals that may be vectors in a time. If the signal and noise are jointly Gaussian, the Kalman filter is an optimal MMSE estimator, else is an optimal LMMSE estimator.

We always assume that the parameter is the *vector Gauss-Markov process*

$$\boldsymbol{\theta}_n = \mathbf{A}_n \boldsymbol{\theta}_{n-1} + \mathbf{B}_n \mathbf{u}_n \tag{3.111}$$

where $\mathbf{A}_n$ is a $p \times p$ nonsingular state-transition matrix with eigenvalues smaller than one in magnitude, $\mathbf{u}_n$ is an r-dimensional vector being a Gaussian random variable uncorrelated over time, named as *driving noise*, with zero mean and covariance matrix $\mathbf{Q}_n$

$$\mathrm{E}\left[\mathbf{u}_{n+m}\mathbf{u}_n^T\right] = \begin{cases} \mathbf{0} & n \neq m \\ \mathbf{Q}_n & n = m \end{cases} \tag{3.112}$$

and $\mathbf{B}_n$ is a $p \times r$ matrix. The initial value of the parameter is assumed to be Gaussian $\boldsymbol{\theta}_{-1} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}}, \mathbf{C}_{\boldsymbol{\theta}}\right)$ and independent of $\mathbf{u}_n$ for $n \geq 0$. The equation (3.111) is termed as the *state-space model*. The state-space model describes how the parameter of our interest $\boldsymbol{\theta}_n$ randomly evolves over time based on its previous realization $\boldsymbol{\theta}_{n-1}$. The larger the elements of $\mathbf{Q}_n$, the more rapidly $\boldsymbol{\theta}_n$ changes.

If $\mathbf{x}_n$ is an $M \times 1$ observation vector modeled by the Bayesian linear model at time n

$$\mathbf{x}_n = \mathbf{H}_n \boldsymbol{\theta}_n + \mathbf{w}_n \tag{3.113}$$

where $\mathbf{H}_n$ is $M \times p$ known matrix, $\mathbf{w}_n$ is $M \times 1$ zero mean Gaussian observation vector, $\mathbf{w}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_n)$, uncorrelated over time with covariance matrix $\mathbf{C}_n$

$$\mathrm{E}\left[\mathbf{w}_{n+m}\mathbf{w}_n^T\right] = \begin{cases} \mathbf{0} & n \neq m \\ \mathbf{C}_n & n = m \end{cases} \tag{3.114}$$

the sequential MMSE estimator of $\boldsymbol{\theta}_n$, *named as Kalman filter*, can be summarized by the following recursion, denoting $\tilde{\mathbf{q}}_n$ prediction of random vector $\mathbf{q}_n$ at time n and $\tilde{\mathbf{C}}_{\mathbf{q}_n}$ covariance matrix of the prediction:

Prediction:

$$\tilde{\boldsymbol{\theta}}_n = \mathbf{A}_n \hat{\boldsymbol{\theta}}_{n-1}. \tag{3.115}$$

Minimum Prediction MSE Matrix:

$$\tilde{\mathbf{M}}_n = \mathbf{A}_n \mathbf{M}_n \mathbf{A}_n^T + \mathbf{B}_n \mathbf{Q}_n \mathbf{B}_n^T. \tag{3.116}$$

Kalman Gain Matrix:

$$\mathbf{K}_n = \tilde{\mathbf{M}}_n \mathbf{H}_n^T \left(\mathbf{H}_n \tilde{\mathbf{M}}_n \mathbf{H}_n^T + \mathbf{C}_n\right)^{-1}. \tag{3.117}$$

Correction:

$$\hat{\boldsymbol{\theta}}_n = \tilde{\boldsymbol{\theta}}_n + \mathbf{K}_n \left(\mathbf{x}_n - \mathbf{H}_n \tilde{\boldsymbol{\theta}}_n\right). \tag{3.118}$$

Minimum MSE Matrix:

$$\mathbf{M}_n = \left(\mathbf{I} - \mathbf{K}_n \mathbf{H}_n\right) \tilde{\mathbf{M}}_n. \tag{3.119}$$

The recursion is initialized with $\hat{\boldsymbol{\theta}}_{-1} = \boldsymbol{\mu}_{\boldsymbol{\theta}}$, $\tilde{\mathbf{M}}_{-1} = \mathbf{C}_{\boldsymbol{\theta}}$.

If the dimension of the state-space vector p is less then the observation vector M, a more efficient form of the Kalman filter can be obtained. It is referred to as *information form.* The number of flops for the filter (3.115)-(3.119) grow cubicly with the dimension of the observation vector M due to the inversion in (3.117). The informative form, thanks to *matrix inversion lemma* applied to (3.117), changes the number of flops so that it depends on p cubicly and on M quadraticly. The matrix

inversion lemma is in (A.1). The number of flops for the standard form of the Kalman filter is, using the expressions for complexity of matrix algebra in Table A.1,

$$\begin{aligned}\mathcal{O}_{\text{KF, Stand.}}(p,\, M,\, r) &= \frac{2}{3}M^3 + 5p^3 + 4M^2p + 4Mp^2 + 2p^2r + M^2 && (3.120)\\ &\quad -3p^2 - 2Mp + pr + M + p - 1 && (3.121)\end{aligned}$$

where p is the size of the state-space vector $\boldsymbol{\theta}_n$, M is the size of the observation vector $\mathbf{x}_n$, and r is the size of the driving noise vector $\mathbf{u}_n$. For the information form of the KF, we get the complexity of

$$\begin{aligned}\mathcal{O}_{\text{KF, Inform.}}(p,\, M,\, r) &= \frac{17}{3}p^3 + 10Mp^2 + 2M^2p + 2p^2r && (3.122)\\ &\quad -p^2 - M^2 - Mp + pr + 2M + p - 1. && (3.123)\end{aligned}$$

It should be noted that matrices $\mathbf{A}$, $\mathbf{B}$ of the state-space model are usually sparse so that the complexity of the Kalman gain matrix, correction and minimum MSE matrix dominate.

3.3.3 Extended Kalman Filter (EKF)

It is sometimes the case that the signal or the state-space model are not linear. In spite of this fact, a suboptimal version of the Kalman filter can be constructed using first order Taylor approximation. Such filter is referred to as *extended Kalman filter* (EKF). Its performance is strictly dependent upon the accuracy of the linearization.

Suppose the state-space and the signal model

$$\begin{aligned}\boldsymbol{\theta}_n &= \mathbf{a}_n(\boldsymbol{\theta}_{n-1}) + \mathbf{B}\mathbf{u}_n && (3.124)\\ \mathbf{x}_n &= \mathbf{h}_n(\boldsymbol{\theta}_n) + \mathbf{w}_n && (3.125)\end{aligned}$$

where $\mathbf{a}_n$ is a p-dimensional function and $\mathbf{h}_n$ is an M-dimensional function. The EKF recursion is then similar to (3.115)-(3.119) except for prediction and correction:

Prediction:

$$\tilde{\boldsymbol{\theta}}_n = \mathbf{a}_n\left(\hat{\boldsymbol{\theta}}_{n-1}\right). \tag{3.126}$$

Correction:

$$\hat{\boldsymbol{\theta}}_n = \tilde{\boldsymbol{\theta}}_n + \mathbf{K}_n\left(\mathbf{x}_n - \mathbf{h}_n\left(\tilde{\boldsymbol{\theta}}_n\right)\right). \tag{3.127}$$

In (3.116), (3.117), (3.119), we substitute

$$\mathbf{A}_n = \left.\frac{\partial \mathbf{a}_n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\hat{\boldsymbol{\theta}}_{n-1}} \tag{3.128}$$

$$\mathbf{H}_n = \left.\frac{\partial \mathbf{h}_n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\tilde{\boldsymbol{\theta}}_n}. \tag{3.129}$$

3.4 Posterior Cramer-Rao Lower Bound (PCRLB)

Cramer-Rao lower bound (CRLB) can be found with respect to parameter $\boldsymbol{\theta}$ as a random vector. The covariance matrix is then "averaged over all $\boldsymbol{\theta}$", whereas CRLB for a classical estimator depends on particular value of $\boldsymbol{\theta}$. We can define the Bayesian Fisher information matrix as

$$[\mathbf{J}_B]_{i,j} = -\mathrm{E}_{\mathbf{x},\boldsymbol{\theta}}\left[\frac{\partial \ln p(\mathbf{x},\boldsymbol{\theta})}{\partial\theta_i \partial\theta_j}\right]. \tag{3.130}$$

It can be shown that the covariance matrix is always greater or equal than the inverse of the Bayesian Fisher information matrix [102]

$$\mathbf{C}_{\hat{\boldsymbol{\theta}}} \triangleq \mathrm{E}_{\mathbf{x},\boldsymbol{\theta}}\left[\left(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}\right)\left(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}\right)^T\right] \geq \mathbf{J}_B^{-1}. \tag{3.131}$$

Bayesian estimator is said to be *efficient* when it attains PCRLB. It can be derived that $\mathbf{J}_B$ can be decomposed into an addition of mean classical Fisher information matrix $\mathbf{J}$, named as *data matrix* $\mathbf{J}_D$, and the *prior matrix* $\mathbf{J}_P$

$$\mathbf{J}_B = \mathbf{J}_D + \mathbf{J}_P \tag{3.132}$$

$$[\mathbf{J}_D]_{i,j} = \mathrm{E}_{\boldsymbol{\theta}}\left[-\mathrm{E}_{\mathbf{x}|\boldsymbol{\theta}}\left[\frac{\partial^2 \ln p(\mathbf{x}|\boldsymbol{\theta})}{\partial\theta_i\partial\theta_j}\right]\right] \tag{3.133}$$

$$[\mathbf{J}_P]_{i,j} = -\mathrm{E}_{\boldsymbol{\theta}}\left[\frac{\partial^2 \ln p(\boldsymbol{\theta})}{\partial\theta_i\partial\theta_j}\right]. \tag{3.134}$$

The data matrix $\mathbf{J}_D$ represents a contribution of data, whereas prior matrix $\mathbf{J}_P$ is a contribution of the prior knowledge about parameter $\boldsymbol{\theta}$.

If we consider the incoming data over time, the dimensionality of $\mathbf{J}_B$ grows. To prevent this, a recursive algorithm for determining the PCRLB can be derived. Let us define the following notation

$$\boldsymbol{\theta}_{1:n} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n\} \tag{3.135}$$

$$\mathbf{x}_{1:n} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \tag{3.136}$$

where $\boldsymbol{\theta}_n$ is $p \times 1$ vector and $\mathbf{x}_n$ is $M \times 1$ vector. Next, we assume a generalized state-space and signal model of (3.124), (3.125)

$$\begin{aligned} \boldsymbol{\theta}_n &= \mathbf{a}_n\left(\boldsymbol{\theta}_{n-1},\, \mathbf{u}_n\right) && (3.137)\\ \mathbf{x}_n &= \mathbf{h}_n\left(\mathbf{x}_n,\, \mathbf{w}_n\right) && (3.138) \end{aligned}$$

where $\mathbf{h}_n$, $\mathbf{a}_n$, $p\left(\boldsymbol{\theta}_0\right)$ are known and $\mathbf{u}_n$, $\mathbf{w}_n$ independent. The PCRLB for the filtered output at time n is then defined as

$$\mathbf{C}_{\boldsymbol{\theta}_n} = \mathrm{E}_{\mathbf{x}_n,\boldsymbol{\theta}_n}\left[\left(\hat{\boldsymbol{\theta}}_n\left(\mathbf{x}_{1:n}\right) - \boldsymbol{\theta}_n\right)\left(\hat{\boldsymbol{\theta}}\left(\mathbf{x}_{1:n}\right) - \boldsymbol{\theta}_n\right)^T\right] \geq \mathbf{J}_n^{-1}. \tag{3.139}$$

The estimator is said to be *statistically efficient* if it attains the bound. The recursive computation of the PCRLB can be summarized by the set of the following equations

$$\mathbf{J}_{n+1} = \mathbf{D}_n^{22} - \mathbf{D}_n^{21}\left(\mathbf{J}_n + \mathbf{D}_n^{11}\right)^{-1}\mathbf{D}_n^{12} \tag{3.140}$$

where

$$\begin{aligned} \mathbf{D}_n^{11} &= \mathrm{E}_{\boldsymbol{\theta}_n,\boldsymbol{\theta}_{n+1}}\left[-\triangle_{\boldsymbol{\theta}_n}^{\boldsymbol{\theta}_n}\ln p\left(\boldsymbol{\theta}_{n+1}|\boldsymbol{\theta}_n\right)\right] && (3.141)\\ \mathbf{D}_n^{12} &= \mathrm{E}_{\boldsymbol{\theta}_n,\boldsymbol{\theta}_{n+1}}\left[-\triangle_{\boldsymbol{\theta}_n}^{\boldsymbol{\theta}_{n+1}}\ln p\left(\boldsymbol{\theta}_{n+1}|\boldsymbol{\theta}_n\right)\right] && (3.142)\\ \mathbf{D}_n^{21} &= \mathrm{E}_{\boldsymbol{\theta}_n,\boldsymbol{\theta}_{n+1}}\left[-\triangle_{\boldsymbol{\theta}_{n+1}}^{\boldsymbol{\theta}_n}\ln p\left(\boldsymbol{\theta}_{n+1}|\boldsymbol{\theta}_n\right)\right] && (3.143)\\ &= \left(\mathbf{D}_n^{12}\right)^T && (3.144)\\ \mathbf{D}_n^{22} &= \mathrm{E}_{\boldsymbol{\theta}_n,\boldsymbol{\theta}_{n+1}}\left[-\triangle_{\boldsymbol{\theta}_{n+1}}^{\boldsymbol{\theta}_{n+1}}\ln p\left(\boldsymbol{\theta}_{n+1}|\boldsymbol{\theta}_n\right)\right] && (3.145)\\ &+ \mathrm{E}_{\boldsymbol{\theta}_{n+1},\mathbf{x}_{n+1},}\left[-\triangle_{\boldsymbol{\theta}_{n+1}}^{\boldsymbol{\theta}_{n+1}}\ln p\left(\mathbf{x}_{n+1}|\boldsymbol{\theta}_{n+1}\right)\right] && (3.146)\\ \mathbf{J}_0 &= \mathrm{E}_{\boldsymbol{\theta}_0}\left[-\triangle_{\boldsymbol{\theta}_0}^{\boldsymbol{\theta}_0}\ln p\left(\boldsymbol{\theta}_0\right)\right] && (3.147) \end{aligned}$$

where $\triangle_{\boldsymbol{\theta}_1}^{\boldsymbol{\theta}_2} f\left(\boldsymbol{\theta}_1,\, \boldsymbol{\theta}_2\right) = \nabla_{\boldsymbol{\theta}_1}\left[\nabla_{\boldsymbol{\theta}_2}^T f\left(\boldsymbol{\theta}_1,\, \boldsymbol{\theta}_2\right)\right]$ is the second order partial derivative of function $f\left(\boldsymbol{\theta}_1,\, \boldsymbol{\theta}_2\right)$.

If the state-space model is linear and the useful signal is embedded in additive Gaussian noise

$$\begin{aligned} \boldsymbol{\theta}_n &= \mathbf{A}_n\boldsymbol{\theta}_{n-1} + \mathbf{u}_n && (3.148)\\ \mathbf{x}_n &= \mathbf{h}_n\left(\boldsymbol{\theta}_n\right) + \mathbf{w}_n && (3.149) \end{aligned}$$

the recursion can be simplified as

$$\begin{aligned}
\mathbf{J}_0 &= \mathbf{Q}_0^{-1} && (3.150)\\
\mathbf{D}_n^{11} &= \mathbf{A}_n^T \mathbf{Q}_n^{-1} \mathbf{A}_n && (3.151)\\
\mathbf{D}_n^{12} &= -\mathbf{A}_n^T \mathbf{Q}_n^{-1} && (3.152)\\
\mathbf{D}_n^{22} &= \mathbf{Q}_n^{-1} + \mathrm{E}_{\boldsymbol{\theta}_{n+1}} \left[\mathbf{H}_{n+1}^T \mathbf{C}_{n+1}^{-1} \mathbf{H}_{n+1} \right]. && (3.153)
\end{aligned}$$

Based on the prior knowledge about $\boldsymbol{\theta}_{n+1}$ from its previous estimate, the ensemble in (3.153) can be approximated using Monte Carlo integration, see Section 4.3 for details. In many cases, $\mathbf{Q}_n$ is singular and cannot be inverted. Using matrix inversion lemma A.1, the recursion can be rewritten to the following form

$$\mathbf{J}_{n+1}^{-1} = \mathbf{X}_n \left(\mathbf{X}_n + \mathbf{Y}_n \right)^{-1} \mathbf{Y}_n \tag{3.154}$$

where $\mathbf{X}_n = \left(\mathrm{E}_{\boldsymbol{\theta}} \left[\mathbf{H}_{n+1}^T \mathbf{C}_{n+1}^{-1} \mathbf{H}_{n+1} \right] \right)^{-1}$ and $\mathbf{Y}_n = \mathbf{Q}_n + \mathbf{A}_n \mathbf{J}_n^{-1} \mathbf{A}_n^T$.

Chapter 4

Nonlinear Bayesian Filters

In this chapter, we briefly introduce the nonlinear Bayesian filters that have already been investigated in connection with the PVT estimation in GNSS. These include the *unscented Kalman filter*, the *grid-based filter* and the *particle filter*. While the complexity of the unscented Kalman filter is said to be comparable to the EKF [22, 23], grid-based methods and particle filters are mentioned to introduce an overhead due to a large number of samples needed for PDF representation [26, 27].

We assume Markovian state-space model

$$p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1},\dots\boldsymbol{\theta}_0\right) = p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}\right) \tag{4.1}$$

meaning that no additional information about the current state is contained in the old states but the previous state. Next, we suppose that observations depend only on the current state

$$p\left(\mathbf{x}_n|\boldsymbol{\theta}_{0:n}\right) = p\left(\mathbf{x}_n|\boldsymbol{\theta}_n\right). \tag{4.2}$$

Note that the KF, EKF follow these assumptions as well.

The following equations then summarize the Bayesian recursion for a general PDF representation fulfilling our assumptions [25, 26].

Prediction:

$$p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n-1}\right) = \int p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}\right) p\left(\boldsymbol{\theta}_{n-1}|\mathbf{x}_{1:n-1}\right) \mathrm{d}\boldsymbol{\theta}_{n-1}. \tag{4.3}$$

Update:

$$p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right) = \frac{p\left(\mathbf{x}_n|\boldsymbol{\theta}_n\right) p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n-1}\right)}{\int p\left(\mathbf{x}_n|\boldsymbol{\theta}_n\right) p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n-1}\right) \mathrm{d}\boldsymbol{\theta}_n}. \tag{4.4}$$

The MMSE estimate can then be obtained as

$$\hat{\boldsymbol{\theta}}_n = \mathrm{E}\left[\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right] \tag{4.5}$$

$$= \int \boldsymbol{\theta}_n p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right) \mathrm{d}\boldsymbol{\theta}_n. \tag{4.6}$$

and the vector MAP estimate as

$$\hat{\boldsymbol{\theta}}_n = \mathrm{argmax}_{\boldsymbol{\theta}_n}\, p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right). \tag{4.7}$$

4.1 Unscented Kalman Filter (UKF)

The unscented Kalman filter [22] is based on so called *unscented transform* (UT) which aims to compute the statistics of a random vector at the output of a nonlinear system. It deterministically selects a set of points of the random vector of our interest, named as *sigma points*, and propagates the set through the nonlinear transformation. Based on the output points, the UT proposes a mean and a covariance matrix. In the *unscented Kalman filter* (UKF), the estimate of the mean of the posterior PDF is used to approximate the MMSE estimate and the covariance matrix measures the quality of the estimate.

4.1.1 Unscented Transform (UT)

Assume that $\mathbf{a}$ is a random variable with mean $\boldsymbol{\mu}_\mathbf{a}$ and covariance matrix $\mathbf{C}_\mathbf{a}$. Suppose $\mathbf{a}$ propagates through, generally, a nonlinear system $\mathbf{b} = \mathbf{g}\left(\mathbf{a}\right)$ where $\mathbf{g}$ is a vector function describing that system. We represent $\mathbf{a}$ with $2n_\mathbf{a} + 1$ sigma points $\{\mathcal{A}_i\}_{i=0}^{2n_\mathbf{a}}$ selected deterministically as

$$\mathcal{A}_0 = \boldsymbol{\mu}_\mathbf{a} \tag{4.8}$$

$$\mathcal{A}_i = \boldsymbol{\mu}_\mathbf{a} + \left[\sqrt{\left(n_\mathbf{a} + \kappa\right)\mathbf{C}_\mathbf{a}}\right]_{:,i}, \quad i = 1, \ldots, n_\mathbf{a} \tag{4.9}$$

$$\mathcal{A}_i = \boldsymbol{\mu}_\mathbf{a} - \left[\sqrt{\left(n_\mathbf{a} + \kappa\right)\mathbf{C}_\mathbf{a}}\right]_{:,i}, \quad i = n_\mathbf{a} + 1, \ldots, 2n_\mathbf{a} \tag{4.10}$$

where $\mathbf{L} = \sqrt{\mathbf{A}}$ denotes such a matrix $\mathbf{L}$ that $\mathbf{A} = \mathbf{L}\mathbf{L}^T$. Matrix $\mathbf{L}$ can be obtained using Choleski decomposition if $\mathbf{A}$ is nonsingular. To select κ, commonly the following expression is employed [22]

$$\kappa = \kappa_1^2\left(n_\mathbf{a} + \kappa_2\right) - n_\mathbf{a} \tag{4.11}$$

where $\kappa_1 = 10^{-3}$, $\kappa_2 = 0$. The associated weight points are defined as

$$\mathcal{W}_0 = \frac{\kappa}{\kappa + n_{\mathbf{a}}} \tag{4.12}$$

$$\mathcal{W}_i = \frac{0.5}{\kappa + n_{\mathbf{a}}}, \quad i = 1, \ldots, 2n_{\mathbf{a}} \ . \tag{4.13}$$

The weights are thus normalized so it holds that

$$\sum_{i=0}^{2n_{\mathbf{a}}} \mathcal{W}_i = 1. \tag{4.14}$$

To get the output statistical description, we calculate the sigma points representing the output random variable $\mathbf{b}$ as

$$\mathcal{B}_i = \mathbf{g}\left(\mathcal{A}_i\right), \quad i = 0, \ldots, 2n_{\mathbf{a}} + 1 \ . \tag{4.15}$$

The mean $\boldsymbol{\mu}_{\mathbf{b}}$ and the covariance matrix $\mathbf{C}_{\mathbf{b}}$ of the output variable is then estimated

$$\boldsymbol{\mu}_{\mathbf{b}} = \sum_{i=0}^{2n_{\mathbf{a}}} \mathcal{W}_i \mathcal{B}_i \tag{4.16}$$

$$\mathbf{C}_{\mathbf{b}} = \sum_{i=0}^{2n_{\mathbf{a}}} \mathcal{W}_i \left(\mathcal{B}_i - \boldsymbol{\mu}_{\mathbf{b}}\right)\left(\mathcal{B}_i - \boldsymbol{\mu}_{\mathbf{b}}\right)^T. \tag{4.17}$$

4.1.2 Unscented Kalman Filter (UKF)

The UKF assumes that the posterior PDF is Gaussian concentrated about the MMSE estimate with MSE matrix as the covariance matrix

$$p\left(\boldsymbol{\theta}_{n-1} | \mathbf{x}_{1:n-1}\right) = \mathcal{N}_{\boldsymbol{\theta}_{n-1}}\left(\hat{\boldsymbol{\theta}}_{n-1},\, \mathbf{M}_{n-1}\right). \tag{4.18}$$

The distribution is approximated as a set sigma points with the corresponding weights $\left\{\mathcal{Z}^i_{n-1},\, \mathcal{W}^i_{n-1}\right\}_{i=0}^{2n_{\boldsymbol{\theta}}}$ obtained by the UT. The UKF equations follow [22]

$$\tilde{\mathcal{Z}}^i_n = \mathbf{a}_n\left(\mathcal{Z}^i_{n-1}\right) \tag{4.19}$$

$$\tilde{\boldsymbol{\theta}}_n = \sum_{i=0}^{2n_{\boldsymbol{\theta}}} \mathcal{W}^i_{n-1} \tilde{\mathcal{Z}}^i_n \tag{4.20}$$

$$\tilde{\mathbf{M}}_n = \mathbf{Q}_n + \sum_{i=0}^{2n_{\boldsymbol{\theta}}} \mathcal{W}^i_{n-1} \left(\tilde{\mathcal{Z}}^i_n - \tilde{\boldsymbol{\theta}}_n\right)\left(\tilde{\mathcal{Z}}^i_n - \tilde{\boldsymbol{\theta}}_n\right)^T \tag{4.21}$$

$$\hat{\boldsymbol{\theta}}_n = \tilde{\boldsymbol{\theta}}_n + \mathbf{K}_n\left(\mathbf{x}_n - \tilde{\mathbf{x}}_n\right) \tag{4.22}$$

$$\mathbf{M}_n = \tilde{\mathbf{M}}_n - \mathbf{K}_n \mathbf{S}_n \mathbf{K}_n^T \tag{4.23}$$

where

$$\tilde{\mathbf{x}}_n = \sum_{i=0}^{2n_{\boldsymbol{\theta}}} \mathcal{W}_{n-1}^i \mathbf{h}_n\left(\tilde{\mathcal{Z}}_n^i\right) \tag{4.24}$$

$$\mathbf{K}_n = \mathbf{C}_{\tilde{\boldsymbol{\theta}}_n \tilde{\mathbf{x}}_n} \mathbf{S}_n^{-1} \tag{4.25}$$

$$\mathbf{S}_n = \mathbf{C}_{\tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n} + \mathbf{C}_{\mathbf{w}_n} \tag{4.26}$$

where

$$\mathbf{C}_{\tilde{\boldsymbol{\theta}}_n \tilde{\mathbf{x}}_n} = \sum_{i=0}^{2n_{\boldsymbol{\theta}}} \mathcal{W}_{n-1}^i \left(\tilde{\mathcal{Z}}_n^i - \tilde{\boldsymbol{\theta}}_n\right)\left(\mathbf{h}_n\left(\tilde{\mathcal{Z}}_n^i\right) - \tilde{\mathbf{x}}_n\right)^T \tag{4.27}$$

$$\mathbf{C}_{\tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n} = \sum_{i=0}^{2n_{\boldsymbol{\theta}}} \mathcal{W}_{n-1}^i \left(\mathbf{h}_n\left(\tilde{\mathcal{Z}}_n^i\right) - \tilde{\mathbf{x}}_n\right)\left(\mathbf{h}_n\left(\tilde{\mathcal{Z}}_n^i\right) - \tilde{\mathbf{x}}_n\right)^T. \tag{4.28}$$

4.2 Grid-Based Methods

The grid-based methods adopt the approach where the posterior PDF is sampled at deterministically selected points and assign weight to these points equal to their PDF at these points. Suppose N_s selected values of the parameter to be estimated $\left\{\boldsymbol{\theta}_{n-1}^i\right\}_{i=1}^{N_s}$, then

$$\Pr\left[\boldsymbol{\theta}_{n-1} = \boldsymbol{\theta}_{n-1}^i | \mathbf{x}_{1:n-1}\right] \triangleq w_{n-1}^i \tag{4.29}$$

so that we approximate the posterior distribution at time $n-1$ as

$$p\left(\boldsymbol{\theta}_{n-1} | \mathbf{x}_{1:n-1}\right) = \sum_{i=1}^{N_s} w_{n-1}^i \delta\left(\boldsymbol{\theta}_{n-1} - \boldsymbol{\theta}_{n-1}^i\right). \tag{4.30}$$

The prior distribution is then approximated as

$$p\left(\boldsymbol{\theta}_n | \mathbf{x}_{1:n-1}\right) = \sum_{i=1}^{N_s} \tilde{w}_n^i \delta\left(\boldsymbol{\theta}_n - \boldsymbol{\theta}_n^i\right) \tag{4.31}$$

where

$$\tilde{w}_n^i \triangleq \sum_{j=1}^{N_s} w_{n-1}^j p\left(\boldsymbol{\theta}_n^i | \boldsymbol{\theta}_{n-1}^j\right). \tag{4.32}$$

To finish the recursion the posterior distribution is estimated as

$$p\left(\boldsymbol{\theta}_n | \mathbf{x}_{1:n}\right) = \sum_{i=1}^{N_s} w_n^i \delta\left(\boldsymbol{\theta}_n - \boldsymbol{\theta}_n^i\right) \tag{4.33}$$

where

$$w_n^i \triangleq \frac{\tilde{w}_n^i p\left(\mathbf{x}_n|\boldsymbol{\theta}_n^i\right)}{\sum_{j=1}^{N_s} \tilde{w}_n^j p\left(\mathbf{x}_n|\boldsymbol{\theta}_n^j\right)}. \tag{4.34}$$

The samples $\left\{\boldsymbol{\theta}_n^i\right\}_{i=1}^{N_s}$ can be taken the same as $\left\{\boldsymbol{\theta}_{n-1}^i\right\}_{i=1}^{N_s}$. However, if the samples at time $n-1$ represent a concentrated PDF and in the next step the mean of the PDF will change abruptly, the estimator might perform poorly. A resampling technique should be employed. We will reveal this topic in the next section.

4.3 Particle Filtering

The lack of diversity when representing a PDF by deterministically selected samples may be overcome by generating the samples randomly with respect to its PDF. Additionally, Monte Carlo methods can then be suitably fitted to the marginalization and evaluation of the ensembles. This forms the basic principle of *particle filtering* [24–26]. Occasionally, the samples can be randomly generated directly from its PDF. Hence, the concept of *importance sampling* is introduced to generate the samples from a proposing density that must be somehow related to the density we want to sample from. Such particles, after several iterations, will anyway either exhibit low weight or collapse to a single point. *Resampling techniques* have been proposed in the literature to deal with this problem [26]. It should be noted that particle filters perform well when the signal is not "too deterministic". In that case, the diversity of the particles is difficult to obtain and Monte Carlo methods fail. For state-space models that can be partitioned to a linear and nonlinear set of equations, *Rao-Blackwellization* may be adopted yielding "a hybrid particle and Kalman filter" with reduced complexity and with lower risk of divergence. Another advantage of particle filters appearing in the literature is their high level of parallelism; we will try to touch this topic in the thesis.

In this section, we will first introduce the Monte Carlo integration principle, then explain how the importance sampling methods can be used to generate samples and follow with sequential importance sampling methods bringing us to the recursion of a particle filter. Next, we discuss resampling and deliver an example of a particle filter where the importance density equals the prior density, the *Bootstrap filter*. This is an easy way of implementing a particle filter, since the prior distribution may be assumed Gaussian for our application. The drawback is that the current observed data are not regarded when generation of new particles. Again, this might result in poor performance, if the prior density is not matched well. Therefore, attention must be paid to the selection of the importance density.

4.3.1 Monte Carlo Integration

The Monte Carlo integration principle can be incorporated to recursively estimate $p(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n})$ or $p(\boldsymbol{\theta}_n|\mathbf{x}_{1:n})$ and its associated expectations. If $g_n(\boldsymbol{\theta}_{0:n})$ is a function of $\boldsymbol{\theta}_{0:n}$, the expectation of that function $I(g_n)$ is

$$I(g_n) = \mathrm{E}[g_n(\boldsymbol{\theta}_{0:n})|\mathbf{x}_{1:n}] \tag{4.35}$$

$$= \int g_n(\boldsymbol{\theta}_{0:n}) p(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}) \mathrm{d}\boldsymbol{\theta}_{0:n}. \tag{4.36}$$

Assume we can generate N_s IID samples $\{\boldsymbol{\theta}^i_{0:n}\}_{i=1}^{N_s}$ from $p(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n})$, meaning these values attain the PDF, the estimates of the distribution and its expectation are

$$p_s(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}) = \frac{1}{N_s}\sum_{i=1}^{N_s}\delta(\boldsymbol{\theta}_{0:n} - \boldsymbol{\theta}^i_{0:n}) \tag{4.37}$$

$$I_s(g_n) = \int g_n(\boldsymbol{\theta}_{0:n}) p_s(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}) \mathrm{d}\boldsymbol{\theta}_{0:n} \tag{4.38}$$

$$= \frac{1}{N_s}\sum_{i=1}^{N_s} g_n(\boldsymbol{\theta}^i_{0:n}). \tag{4.39}$$

We add lower index s to denote that we approximate the PDF or expectation with samples. Strictly speaking, we approximate the PDF with discrete values and we hence obtain a sum of Dirac pulses. If the PDF is continuous, the values are different to what we get. Since we are interested in evaluating the expectation, such approximation is acceptable.

It can be proved that the estimate of the expectation is asymptotically unbiased [24]

$$\Pr[\lim_{N_s\to\infty}(I_s(g_n) = I(g_n))] = 1. \tag{4.40}$$

If the variance of the function g_n converges

$$\sigma^2_{g_n} = \int (g_n(\boldsymbol{\theta}_{0:n}) - I(g_n))^2 p(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}) \mathrm{d}\boldsymbol{\theta}_{0:n} \tag{4.41}$$

$$= \int g_n^2(\boldsymbol{\theta}_{0:n}) p(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}) \mathrm{d}\boldsymbol{\theta}_{0:n} - I(g_n)^2 < \infty \tag{4.42}$$

the asymptotic distribution of ensemble estimation error is Gaussian with variance equal to the variance of g_n

$$\lim_{N_s\to\infty}\sqrt{N_s}(I_s(g_n) - I(g_n)) \sim \mathcal{N}(0, \sigma^2_{g_n}). \tag{4.43}$$

4.3.2 Importance Sampling

It is not always possible to sample from a distribution, since it can be a complicated and explicitly unknown function. Suppose we can generate samples from a density $\pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)$. The function $\pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)$ is called the *importance density*. It then holds

$$\begin{aligned} I\left(g_n\right) &= \int g_n\left(\boldsymbol{\theta}_{0:n}\right) p\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right) \mathrm{d}\boldsymbol{\theta}_{0:n} && (4.44) \\ &= \int g_n\left(\boldsymbol{\theta}_{0:n}\right) \frac{p\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)}{\pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)} \pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right) \mathrm{d}\boldsymbol{\theta}_{0:n} && (4.45) \\ &= \int g_n\left(\boldsymbol{\theta}_{0:n}\right) \breve{w}\left(\boldsymbol{\theta}_{0:n}\right) \pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right) \mathrm{d}\boldsymbol{\theta}_{0:n} && (4.46) \end{aligned}$$

where we define weights

$$\breve{w}\left(\boldsymbol{\theta}_{0:n}\right) = \frac{p\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)}{\pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)}. \tag{4.47}$$

If we draw N_s samples $\left\{\boldsymbol{\theta}_{0:n}^i\right\}_{i=1}^{N_s}$ from $\pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)$, we can approximate the ensemble as

$$I_s\left(g_n\right) = \frac{1}{N_s} \sum_{i=1}^{N_s} w\left(\boldsymbol{\theta}_{0:n}^i\right) g_n\left(\boldsymbol{\theta}_{0:n}^i\right) \tag{4.48}$$

where $w\left(\boldsymbol{\theta}_{0:n}^i\right)$ are normalized weights of $\breve{w}\left(\boldsymbol{\theta}_{0:n}^i\right)$

$$w\left(\boldsymbol{\theta}_{0:n}^i\right) = \frac{\breve{w}\left(\boldsymbol{\theta}_{0:n}^i\right)}{\sum_{j=1}^{N_s} \breve{w}\left(\boldsymbol{\theta}_{0:n}^j\right)}. \tag{4.49}$$

4.3.3 Sequential Importance Sampling

The disadvantage of the importance sampling (IS) method is that it is not sequential. For every new data the estimation process is started from scratch and the dimensionality grows. The *sequential IS* methods solve this issue. In compliance with the importance density and weights definition it holds that, dropping index s denoting approximation,

$$p\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right) = \frac{1}{N_s} \sum_{i=1}^{N_s} w\left(\boldsymbol{\theta}_{0:n}^i\right) \delta\left(\boldsymbol{\theta}_{0:n} - \boldsymbol{\theta}_{0:n}^i\right) \tag{4.50}$$

where

$$w\left(\boldsymbol{\theta}_{0:n}^i\right) \propto \frac{p\left(\boldsymbol{\theta}_{0:n}^i|\mathbf{x}_{1:n}\right)}{\pi\left(\boldsymbol{\theta}_{0:n}^i|\mathbf{x}_{1:n}\right)}. \tag{4.51}$$

Symbol $\propto$ denotes equality up to a scaling factor. If we choose $\pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)$ to factorize

$$\begin{aligned}\pi\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right) &= \pi\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{0:n-1},\, \mathbf{x}_{1:n}\right)\pi\left(\boldsymbol{\theta}_{0:n-1}|\mathbf{x}_{1:n-1}\right) &(4.52)\\ &= \pi\left(\boldsymbol{\theta}_0\right)\prod_{k=1}^{n}\pi\left(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1},\, \mathbf{x}_{1:k}\right) &(4.53)\end{aligned}$$

then we can draw $\boldsymbol{\theta}_n^i \sim \pi\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{0:n-1},\, \mathbf{x}_{1:n}\right)$. If

$$p\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right) \propto p\left(\mathbf{x}_n|\boldsymbol{\theta}_n\right)p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}\right)p\left(\boldsymbol{\theta}_{0:n-1}|\mathbf{x}_{1:n-1}\right) \tag{4.54}$$

where $p\left(\boldsymbol{\theta}_0\right) = p\left(\mathbf{x}_0|\boldsymbol{\theta}_0\right)$, then

$$\hat{p}\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right) = \sum_{i=1}^{N_s} w_n^i \delta\left(\boldsymbol{\theta}_n - \boldsymbol{\theta}_n^i\right) \tag{4.55}$$

$$w_n^i = w_{n-1}^i \frac{p\left(\mathbf{x}_n|\boldsymbol{\theta}_n^i\right)p\left(\boldsymbol{\theta}_n^i|\boldsymbol{\theta}_{n-1}^i\right)}{\pi\left(\boldsymbol{\theta}_n^i|\boldsymbol{\theta}_{0:n-1}^i,\, \mathbf{x}_{1:n}\right)}. \tag{4.56}$$

The MMSE estimate is then

$$\hat{\boldsymbol{\theta}}_n = \sum_{i=1}^{N_s} w_n^i \boldsymbol{\theta}_n^i. \tag{4.57}$$

The covariance of the estimate is

$$\mathbf{C}_{\boldsymbol{\theta}_n} = \sum_{i=1}^{N_s} w_n^i \left(\boldsymbol{\theta}_n^i - \hat{\boldsymbol{\theta}}_n\right)\left(\boldsymbol{\theta}_n^i - \hat{\boldsymbol{\theta}}_n\right)^T. \tag{4.58}$$

To ensure the asymptotic convergences of the Monte Carlo methods with SIS

$$\Pr\left[\lim_{N_s\to\infty}\hat{p}\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right) - p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right)\right] = 1 \tag{4.59}$$

it must hold that [25]

$$\begin{aligned}\pi_{\text{Set}} &= \left\{\boldsymbol{\theta}_n \text{ is } p\times 1 | \pi\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right) > 0\right\} &(4.60)\\ p_{\text{Set}} &= \left\{\boldsymbol{\theta}_n \text{ is } p\times 1 | p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right) > 0\right\} &(4.61)\\ p_{\text{Set}} &\subseteq \pi_{\text{Set}}. &(4.62)\end{aligned}$$

4.3.4 Resampling

Variance of the importance weights can only increase over time which is called as *degeneracy problem.* After a certain number of sequential steps, values of normalized weights equal only 0 or 1. It is essential to keep particles with significant weight and remove those that hardly contribute to the estimation. Effective number of samples can be defined as [26]

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_s} \left(w_n^i\right)^2} \tag{4.63}$$

based on the variance of the weights. It always holds that $1 \leq \hat{N}_{\text{eff}} \leq N_s$. If N_{eff} is low, new samples should be generated to avoid the degeneracy problem. A practical threshold for the effective number of samples is $2N_s/3$. When resampling with low process noise, a phenomenon where particles that have high weights w_n^i are selected many times should be avoided. This phenomenon is called *sample impoverishment.* For very low process noise, the particles will likely collapse to a single point. Since particle filters are designed as Monte Carlo methods based on statistical diversity, deterministic nature of the measurement process will not ensure proper behavior. A resampling algorithm is described in Algorithm 4.1.

4.3.5 Particle Filter with Bootstrap Filter

The Bootstrap filter is a straightforward implementation of a particle filter where the importance density is selected such that it equals the prior density

$$\pi\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}^i, \mathbf{x}_{1:n}\right) = p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}^i\right). \tag{4.64}$$

It is a suboptimal solution, as current data are not regarded when generating the samples. The algorithm can be summarized as follows [27]

1. Get the new predicted particles by sampling from the prior PDF

$$\boldsymbol{\theta}_n^i \sim p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}^i\right) \tag{4.65}$$

   for initialization $\boldsymbol{\theta}_0^i \sim p\left(\boldsymbol{\theta}_0\right)$, all weights unit.

2. Assign the particle a weight

$$\breve{w}_n^i = p\left(\mathbf{x}_n|\boldsymbol{\theta}_n^i\right) \tag{4.66}$$

   normalize weights

$$w_n^i = \frac{\breve{w}_n^i}{\sum_{j=1}^{N_s} \breve{w}_n^j} \tag{4.67}$$

Algorithm 4.1 Resampling algorithm [26]

Outputs: $\{\boldsymbol{\theta}_n^j\}_{j=1}^{N_s}$, $\{w_n^j\}_{j=1}^{N_s}$
Inputs: $\{\boldsymbol{\theta}_n^i\}_{i=1}^{N_s}$, $\{w_n^i\}_{i=1}^{N_s}$

1. Initialize the CDF: $c_1 = 0$
2. For $i = 2, \ldots, N_s$
   - Construct CDF $c_i = c_{i-1} + w_n^i$
3. Start at the bottom of the CDF: $i = 1$
4. Draw a starting point: $u_1 \sim \mathcal{UD}\,(0,\, N_s^{-1})$
5. For $j = 1, \ldots, N_s$
   - Move along the CDF: $u_j = u_1 + N_s^{-1}\,(j-1)$
   - While $u_j > c_i$
     - $i = i + 1$
   - Assign sample: $\boldsymbol{\theta}_n^j = \boldsymbol{\theta}_n^i$
   - Assign weight: $w_n^j = N_s^{-1}$

estimate the states

$$\hat{\boldsymbol{\theta}}_n = \sum_{i=1}^{N_s} w_n^i \boldsymbol{\theta}_n^i. \tag{4.68}$$

3. *Resampling:* Get N_{eff} from (4.63), if $N_{\text{eff}} < 2N_s/3$, perform resampling - generate new particles $\boldsymbol{\theta}_n^i$ and the corresponding weights w_n^i according to Algorithm 4.1.

Chapter 5

Theory of Factor Graphs and the Sum-Product Algorithm

In this chapter, we briefly define the factor graph and describe the sum-product algorithm. *Factor graph* (FG) is a bipartite graph that represents relations among variables of a system [28, 29]. The system is assumed to be described by a complicated *global function that factors into simpler local functions*, each of which having arguments from a subset of the system variables. The *sum-product algorithm* (SPA) is a generic message-passing (MP) algorithm which operates in a factor graph and *attempts to compute various marginal functions associated with the global function.*

Here, we restrict *factorization and marginalization to multiplication and integration, respectively. The global and local functions will be represented by the probability density functions (PDF).* Although the factor graph framework applies to more general problems, such constraint will make our approach to PVT filtering more illustrative.

Let's assume that global function $p(X) \geq 0$ of K system variables $X = \{x_1, \ldots, x_K\}$ where $\forall k \in \{1, \ldots, K\} : \; x_k \in \mathcal{A}_k : \; \mathcal{A}_k \subset \mathbb{R}$ factors into a product of local functions $\{p_j(X_j) : \; j \in J \wedge p_j(X_j) \geq 0\}$

$$p(X) = \prod_{j \in J} p_j(X_j)$$

for X_j being a subset of the global function arguments $X_j \subset X$ and j denoting the index of the corresponding local function in set J of such indices. Factor graph is a bipartite graph that visualizes the factorization using three types of components: *variable nodes* - each representing the system variable x_k, *factor nodes* - each representing the local function $p_j(X_j)$, and *edges* - connecting the variable nodes x_k and

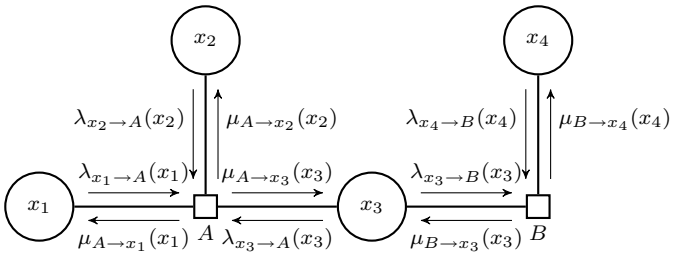


Figure 5.1: Example FG with MP for global function $p(x_1, x_2, x_3, x_4)$ that factors into local functions $p_A(x_1, x_2, x_3)$, $p_B(x_3, x_4)$ in the following manner $p(x_1, x_2, x_3, x_4) = p_A(x_1, x_2, x_3) \cdot p_B(x_3, x_4)$. The direction of the messages is denoted with arrow.

factor nodes $p_j(X_j)$ if and only if $x_k \in X_j$. An example FG where global function $p(x_1, x_2, x_3, x_4)$ factors into local functions $p_A(x_1, x_2, x_3)$, $p_B(x_3, x_4)$ is in Fig. 5.1.

The *marginal function* $p_i(x_i)$ is defined as a summation of the global function $p(x_1, \ldots, x_K)$ over all arguments except x_i, denoted by $\sim x_i$,

$$\begin{aligned} p_i(x_i) &\triangleq \int_{\sim x_i} p(x_1, \ldots, x_K) \mathrm{d}x_1 \ldots \mathrm{d}x_K \\ &\triangleq \int_{\sim x_i} p(X) \mathrm{d}X. \end{aligned}$$

In the given example, e.g. the marginal function $p_3(x_3)$ would be computed as

$$p_3(x_3) = \int_{x_1 \in \mathcal{A}_1} \int_{x_2 \in \mathcal{A}_2} \int_{x_4 \in \mathcal{A}_4} p(x_1, x_2, x_3, x_4) \mathrm{d}x_1 \mathrm{d}x_2 \mathrm{d}x_4.$$

The key property that *multiplication distributes over summation is adopted in a cycle-free FG to distributively compute the resulting marginal function* $p_i(x_i)$ *from marginalized local functions* $\int_{\sim x_i} p_j(X_j) \mathrm{d}X_j$, where $\cap_{j \in J} X_j = x_i \vee \emptyset$,

$$\begin{aligned} p_i(x_i) &= \int_{\sim x_i} p(X) \mathrm{d}X \\ &= \int_{\sim x_i} \prod_{j \in J} p_j(X_j) \mathrm{d}X \\ &= \prod_{j \in J} \left(\int_{\sim x_i} p_j(X_j) \mathrm{d}X_j \right). \end{aligned}$$

In the given example

$$\begin{aligned} p_3(x_3) &= \left(\int_{x_1 \in \mathcal{A}_1} \int_{x_2 \in \mathcal{A}_2} p_A(x_1, x_2, x_3) \mathrm{d}x_1 \mathrm{d}x_2 \right) \\ &\cdot \left(\int_{x_4 \in \mathcal{A}_4} p_B(x_3, x_4) \mathrm{d}x_4 \right). \end{aligned} \tag{5.1}$$

Computation of a *single marginal function in a tree FG* is then a "bottom-up" procedure that starts at the leaf vertices. The leaf vertices send trivial identity functions to their parents which wait for messages from all their children before they start calculation of the local marginal functions. When the marginalization is completed in a vertex, it becomes a child and sends the results to its parents. The algorithm continuous until the target marginal function is obtained.

In our example (Fig. 5.1), in order to get the resulting marginal function $p_3(x_3)$, variable nodes x_1, x_2 first send simple identity messages $\lambda_{x_1 \to A}(x_1)$, $\lambda_{x_2 \to A}(x_2)$ to factor node A, and so sends the variable node x_4 message $\lambda_{x_4 \to B}(x_4)$ to factor node B. The local marginals, the expressions in parenthesis in (5.1), are evaluated and information about the results is send in messages $\mu_{A \to x_3}(x_3)$, $\mu_{B \to x_3}(x_3)$ to variable node x_3. According to (5.1), the resulting marginal function is a product of these local marginals, symbolically,

$$p_3(x_3) = \mu_{A \to x_3}(x_3) \cdot \mu_{B \to x_3}(x_3).$$

We use the term "symbolically", since the massages passed in a FG represent information about a PDF, not necessarily the PDF itself. However, it is instructive to use these messages to denote the corresponding marginal PDFs.

The *sum-product algorithm* is a generalization of the MP algorithm for *efficient calculation of all the marginal functions* associated with the global function. The summary is given in Algorithm 5.1. Messages from factor node to variable node are denoted with μ, whereas messages from variable node to factor node with λ.

In a tree factor graph, the beliefs truly represent the marginals of the global function. However, when *cycles* appear in the graph, vertices infinitely wait for results from each other. A solution to this problem might be to *initialize the messages and let the sum-product algorithm iterate on the FG*. The convergence is mostly difficult to prove. Nonetheless, "appropriate" initial conditions usually succeed.

Algorithm 5.1 The sum-product algorithm

Let x denote a variable node, F a factor node, and $n(x)$, $n(F)$ the sets of neighbors of variable node x and factor node F, respectively. Symbol p_F denotes a local function corresponding to factor node F. The messages sent in a FG are recursively computed according to the following algorithm [28]:
Variable Node to Factor Node:

$$\lambda_{x\to F}(x) = \prod_{G\in n(x)\setminus\{F\}} \mu_{G\to x}(x)$$

Factor Node to Variable Node:

$$\mu_{F\to x}(x) = \int_{\sim x} p_F(X) \prod_{y\in n(F)\setminus\{x\}} \lambda_{y\to F}(y)\mathrm{d}X$$

where X is a set of all arguments of p_F. The marginalized version of the global function with argument x is named *belief* and is evaluated as

$$B(x) = \prod_{G\in n(x)} \mu_{G\to x}(x).$$

Chapter 6

Bayesian Filtering on the FG

In this chapter, we show how a general Bayesian filter can be modeled on the FG and how smoothing and prediction algorithm can be smartly derived from it. We derive the Kalman filter assuming all PDFs Gaussian and explain how the FG with the update rules can be simply modified to approximate a nonlinearity by the extended KF. Then, we introduce a novel iterative filter, named as *scalar Kalman filter*. The filter operates on scalars and features lower complexity and simple arithmetic operations than that of the vector KF. The scalar version of the extended Kalman filter is also introduced.

To create a factor graph for Bayesian filter, we first factorize the posterior PDF of $\boldsymbol{\theta}_{0:n}$ using (4.3) and (4.4)

$$p\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right) \propto p\left(\boldsymbol{\theta}_0\right) \prod_{k=1}^{n} p\left(\mathbf{x}_k|\boldsymbol{\theta}_k\right) p\left(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{k-1}\right). \tag{6.1}$$

The posterior $p\left(\boldsymbol{\theta}_{0:n}|\mathbf{x}_{1:n}\right)$ will be the global function of the FG and the marginalized version of this posterior $p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right)$ is to be calculated recursively as

$$p\left(\boldsymbol{\theta}_n|\mathbf{x}_{1:n}\right) \propto p\left(\mathbf{x}_n|\boldsymbol{\theta}_n\right) \int p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}\right) p\left(\boldsymbol{\theta}_{n-1}|\mathbf{x}_{1:n-1}\right) \mathrm{d}\boldsymbol{\theta}_{n-1}. \tag{6.2}$$

The corresponding FG is depicted in Figure 6.1. The algorithm starts at the left-most vertex. The message representing the PDF of $\boldsymbol{\theta}_0$ is sent to variable node $\boldsymbol{\theta}_0$ where it passes through to the next factor node

$$\mu_{P0\to\boldsymbol{\theta}_0}\left(\boldsymbol{\theta}_0\right) = \lambda_{\boldsymbol{\theta}_0\to P1}\left(\boldsymbol{\theta}_0\right). \tag{6.3}$$

The marginalized message of $\boldsymbol{\theta}_1$ is then sent to variable node $\boldsymbol{\theta}_1$

$$\mu_{P1\to\boldsymbol{\theta}_1}\left(\boldsymbol{\theta}_1\right) = \int p\left(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0\right) \lambda_{\boldsymbol{\theta}_0\to P1}\left(\boldsymbol{\theta}_0\right) \mathrm{d}\boldsymbol{\theta}_0 \tag{6.4}$$

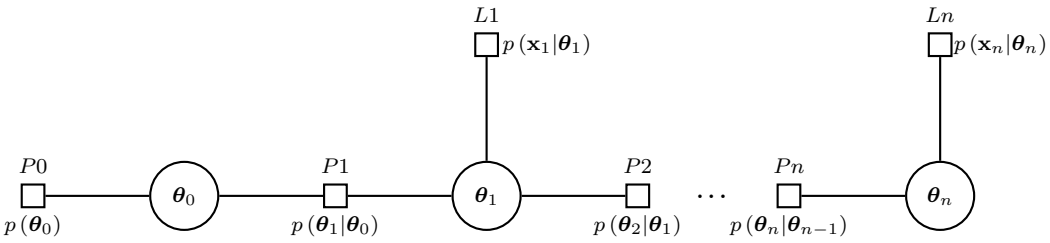


Figure 6.1: Factor graph for Bayesian filtering

where this message is combined with the likelihood message

$$B(\boldsymbol{\theta}_1) = \mu_{P1\to\boldsymbol{\theta}_1}(\boldsymbol{\theta}_1) \cdot \mu_{L1\to\boldsymbol{\theta}_1}(\boldsymbol{\theta}_1) \tag{6.5}$$

and MMSE or MAP estimate is obtained based on the calculated belief

$$\hat{\boldsymbol{\theta}}_1^{\mathrm{MMSE}} = \frac{\int \boldsymbol{\theta}_1 B(\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1}{\int B(\boldsymbol{\theta}_1)\,\mathrm{d}\boldsymbol{\theta}_1} \tag{6.6}$$

$$\hat{\boldsymbol{\theta}}_1^{\mathrm{MAP}} = \mathrm{argmax}_{\boldsymbol{\theta}_1} B(\boldsymbol{\theta}_1). \tag{6.7}$$

This belief is sent to the conditional factor node

$$\lambda_{\boldsymbol{\theta}_1\to P2}(\boldsymbol{\theta}_1) = B(\boldsymbol{\theta}_1). \tag{6.8}$$

and so continuous the algorithm.

Depending on the PDF representation, the marginalizations and combinations may have different forms. Note that the FG does not contain cycles and no iterations are needed. The messages obtained from the current measurement might be sent back on the graph and using the same update rules, *smoothed* estimates can be obtained. When propagating the current messages to the future branches, disconnecting the future likelihood factor nodes and edges, *predictions* can be calculated.

6.1 Kalman Filter on the FG

Kalman filter is a special case of Bayesian filtering with Gaussian PDF representation, linear spate-space model (3.111) and linear measurement equation (3.113), see Section 3.3.2. It is then sufficient to represent the messages with their mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{C}$, since all the derived PDFs on the FG must be Gaussian [49]. We can then derive update rules for both factor and variable nodes depending solely

upon these quantities. The same holds for the belief and the MAP and MMSE estimate.

The update rules for the FG modeling the Kalman filter are summarized in Fig. 6.2. The update rules applied to the FG in Figure 6.1 can be used to derive the Kalman filter, smoother, and predictor. The MMSE and MAP estimates are identical for Gaussian distributions and can be obtained as the mean of the belief which is to be sent from the current variable node $\boldsymbol{\theta}_n$ to the future factor node $P(n+1)$.

The Kalman filter recursion can be derived[1] supposing that the initial value of $\boldsymbol{\theta}$ is distributed as a Gaussian random variable

$$p\left(\boldsymbol{\theta}_0\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_0,\, \mathbf{C}_0\right) \tag{6.9}$$

which mean and covariance matrix is sent from its factor node $P0$ through its variable node to the factor node $P1$

$$\mu_{P0\to\boldsymbol{\theta}_0}\left(\boldsymbol{\theta}_0\right) = \lambda_{\boldsymbol{\theta}_0\to P1}\left(\boldsymbol{\theta}_0\right) = \left\{\boldsymbol{\mu}_0,\, \mathbf{C}_0\right\}. \tag{6.10}$$

Since the state-space model is Gauss-Markov and of the first order (3.111)

$$p\left(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0\right) = \mathcal{N}\left(\mathbf{A}\boldsymbol{\mu}_0,\, \mathbf{B}\mathbf{Q}\mathbf{B}^T\right) \tag{6.11}$$

the message will be

$$\mu_{P1\to\boldsymbol{\theta}_1} = \left\{\mathbf{A}\boldsymbol{\mu}_0,\, \mathbf{A}\mathbf{C}_0\mathbf{A}^T + \mathbf{B}\mathbf{Q}\mathbf{B}^T\right\}. \tag{6.12}$$

Even though the likelihood function has the following PDF

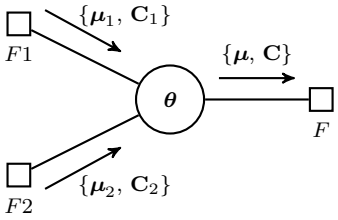
$$p\left(\mathbf{x}|\boldsymbol{\theta}_1\right) = \mathcal{N}_{\mathbf{x}}\left(\mathbf{H}\boldsymbol{\theta}_1,\, \mathbf{C}_1\right) \tag{6.13}$$

the message must be represented by the following mean and covariance matrix, derived in Figure 6.3,

$$\mu_{L1\to\boldsymbol{\theta}_1}\left(\boldsymbol{\theta}_1\right) = \left\{\left(\mathbf{H}^T\mathbf{C}_1^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{C}_1^{-1}\mathbf{x},\, \left(\mathbf{H}^T\mathbf{C}_1^{-1}\mathbf{H}\right)^{-1}\right\}. \tag{6.14}$$

So continuous the algorithm for $n > 1$.

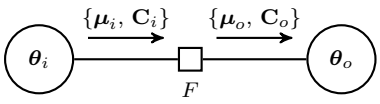
[1] We do not denote dependence of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{Q}$, $\mathbf{H}$ and some other quantities on time for simplicity. The level of variable time generalization applies to this FG model analogically.



$$\mathbf{C} = \left(\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1}\right)^{-1}$$

$$\boldsymbol{\mu} = \mathbf{C}\left(\mathbf{C}_1^{-1}\boldsymbol{\mu}_1 + \mathbf{C}_2^{-1}\boldsymbol{\mu}_2\right)$$

(a)

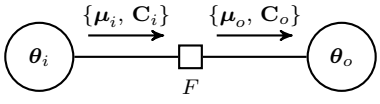


$$\mathbf{C}_o = \mathbf{A}\mathbf{C}_i\mathbf{A}^T + \mathbf{C}_F$$

$$\boldsymbol{\mu}_o = \mathbf{A}\boldsymbol{\mu}_i$$

$$p_F\left(\boldsymbol{\theta}_o|\boldsymbol{\theta}_i\right) = \mathcal{N}_{\boldsymbol{\theta}_o}\left(\mathbf{A}\boldsymbol{\mu}_i,\, \mathbf{C}_F\right)$$

(b)

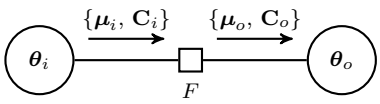


$$\mathbf{C}_o = \mathbf{B}\mathbf{C}_i\mathbf{B}^T$$

$$\boldsymbol{\mu}_o = \mathbf{B}\boldsymbol{\mu}_i$$

$$p_F\left(\boldsymbol{\theta}_o|\boldsymbol{\theta}_i\right) = \delta\left(\boldsymbol{\theta}_o - \mathbf{B}\boldsymbol{\theta}_i\right)$$

(c)

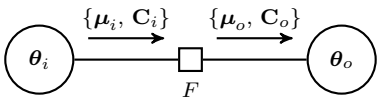


$$\mathbf{C}_o = \left(\mathbf{A}^T\left(\mathbf{C}_i + \mathbf{C}_F\right)^{-1}\mathbf{A}\right)^{-1}$$

$$\boldsymbol{\mu}_o = \mathbf{C}_o\mathbf{A}^T\left(\mathbf{C}_i + \mathbf{C}_F\right)^{-1}\boldsymbol{\mu}_i$$

$$p_F\left(\boldsymbol{\theta}_i|\boldsymbol{\theta}_o\right) = \mathcal{N}_{\boldsymbol{\theta}_i}\left(\mathbf{A}\boldsymbol{\mu}_o,\, \mathbf{C}_F\right)$$

(d)



$$\mathbf{C}_o = \left(\mathbf{B}^T\mathbf{C}_i^{-1}\mathbf{B}\right)^{-1}$$

$$\boldsymbol{\mu}_o = \mathbf{C}_o\mathbf{B}^T\mathbf{C}_i^{-1}\boldsymbol{\mu}_i$$

$$p_F\left(\boldsymbol{\theta}_o|\boldsymbol{\theta}_i\right) = \delta\left(\mathbf{B}\boldsymbol{\theta}_o - \boldsymbol{\theta}_i\right)$$

(e)

Figure 6.2: Update rules for a FG modeling Kalman filter - (a) variable node, (b) factor node with state-space model forward direction, (c) equality factor node forward direction, (d) factor node with state-space model backward direction, (e) equality factor node backward direction. Messages comprise mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{C}$.

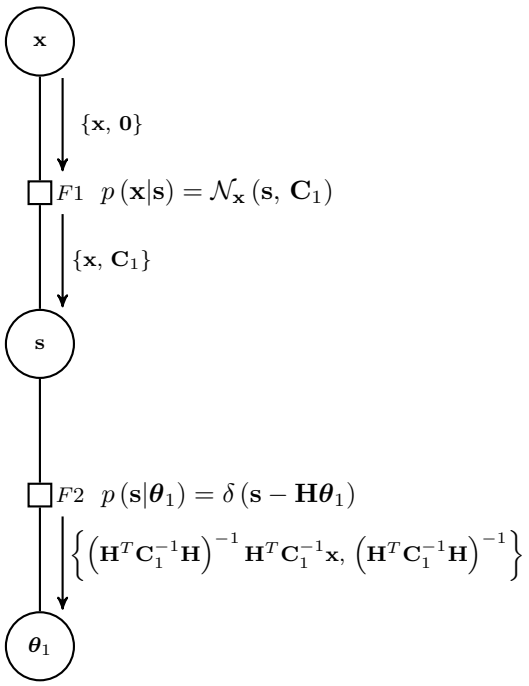


Figure 6.3: Likelihood factor node model for Kalman filter on the FG

6.2 Extended Kalman Filter on the FG

The EKF filter algorithm can be also simply derived on the FG by assuming that all the PDFs are Gaussian and first order Taylor linearization is applied as in Section 3.3.3. The calculation of covariance matrix will remain unchanged in the algorithm compared to the KF algorithm on the FG, however, the mean of the conditional PDF factor node will change so that

$$p\left(\boldsymbol{\theta}_1|\boldsymbol{\theta}_0\right) = \mathcal{N}\left(\mathbf{a}\left(\boldsymbol{\mu}_0\right), \mathbf{B}\mathbf{Q}\mathbf{B}^T\right) \tag{6.15}$$

with message

$$\mu_{P1\to\boldsymbol{\theta}_1} = \left\{\mathbf{a}\left(\boldsymbol{\mu}_0\right), \mathbf{A}\mathbf{C}_0\mathbf{A}^T + \mathbf{B}\mathbf{Q}\mathbf{B}^T\right\}. \tag{6.16}$$

The likelihood message will represent Gaussian PDF with shifted mean by the linearizing value $\mathbf{s} = \mathbf{h}\left(\mathbf{a}\left(\boldsymbol{\mu}_0\right)\right)$

$$\mu_{L1\to\boldsymbol{\theta}_1}\left(\boldsymbol{\theta}_1\right) = \left\{\mathbf{a}\left(\boldsymbol{\mu}_0\right) + \left(\mathbf{H}^T\mathbf{C}_1^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{C}_1^{-1}\left(\mathbf{x}-\mathbf{s}\right), \left(\mathbf{H}^T\mathbf{C}_1^{-1}\mathbf{H}\right)^{-1}\right\}. \tag{6.17}$$

The update rules are identical to those in Figure 6.2.

6.3 Proposed Scalar Iterative Kalman Filter on the FG

To construct a scalar iterative Kalman filter on the FG, we use a similiar approach to [103, 104]. We split all the vector variables into scalars and create a FG for them. Cycles will appear resulting in a certain suboptimality and necessity to investigate convergence and accuracy.

The state-space model for the KF can decomposed as

$$p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}\right) = \int p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}, \mathbf{u}_n\right) p\left(\mathbf{u}_n\right) \mathrm{d}\mathbf{u}_n \tag{6.18}$$

where

$$p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}, \mathbf{u}_n\right) = \prod_{i=1}^{p} \delta\left(\theta_{n,i} - \sum_{j=1}^{p} a_{i,j}\theta_{n-1,j} - \sum_{j=1}^{r} b_{i,j}u_j\right) \tag{6.19}$$

where $a_{i,j} = [\mathbf{A}]_{i,j}$, $b_{i,j} = [\mathbf{B}]_{i,j}$. Suppose $\mathbf{Q}$ being diagonal $\mathbf{Q} = \mathrm{diag}\left(\sigma_{u_1}^2, \ldots, \sigma_{u_r}^2\right)$, we get

$$p\left(\mathbf{u}_n\right) = \prod_{i=1}^{r} \mathcal{N}_{u_i}\left(0, \sigma_{u_i}^2\right). \tag{6.20}$$

If matrix $\mathbf{Q}$ is not diagonal, *singular value decomposition* (SVD) [43, 105] can be adopted to describe vector $\mathbf{u}$ as

$$\mathbf{u} = \mathbf{\Gamma}\boldsymbol{\eta} \tag{6.21}$$

where $\boldsymbol{\eta}$ is $r \times 1$ Gaussian vector with zero mean and diagonal covariance matrix, $\mathbf{\Gamma}$ is $r \times r$ matrix that desirably correlates the elements of vector $\boldsymbol{\eta}$. Then, we would get

$$p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}\right) = \int p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1},\, \boldsymbol{\eta}_n\right) p\left(\boldsymbol{\eta}_n\right) \mathrm{d}\boldsymbol{\eta}_n \tag{6.22}$$

where

$$p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1},\, \boldsymbol{\eta}_n\right) = \prod_{i=1}^{p} \delta\left(\theta_{n,i} - \sum_{j=1}^{p} a_{i,j}\theta_{n-1,j} - \sum_{j=1}^{r} c_{i,j}\eta_j\right) \tag{6.23}$$

where $c_{i,j} = [\mathbf{B}\mathbf{\Gamma}]_{i,j}$. Suppose the variance of ith element of vector $\boldsymbol{\eta}$ is $\sigma^2_{\eta_i}$, then

$$p\left(\boldsymbol{\eta}_n\right) = \prod_{i=1}^{r} \mathcal{N}_{\eta_i}\left(0,\, \sigma^2_{\eta_i}\right). \tag{6.24}$$

We can decompose the likelihood function in a similiar manner

$$p\left(\mathbf{x}_n|\boldsymbol{\theta}_n\right) = \int p\left(\mathbf{x}_n|\boldsymbol{\theta}_n,\, \mathbf{w}_n\right) p\left(\mathbf{w}_n\right) \mathrm{d}\mathbf{w}_n \tag{6.25}$$

where

$$p\left(\mathbf{x}_n|\boldsymbol{\theta}_n,\, \mathbf{w}_n\right) = \prod_{i=1}^{M} \delta\left(x_{n,i} - \sum_{j=1}^{M} h_{i,j}\theta_{n,j} - w_i\right) \tag{6.26}$$

where $h_{i,j} = [\mathbf{H}]_{i,j}$. If we suppose that $\mathbf{w}_n$ has diagonal covariance matrix $\mathbf{C}_n = \mathrm{diag}\left(\sigma^2_{w_1}, \ldots, \sigma^2_{w_M}\right)$, then

$$p\left(\mathbf{w}_n\right) = \prod_{i=1}^{M} \mathcal{N}_{w_i}\left(0,\, \sigma^2_{w_i}\right). \tag{6.27}$$

If $\mathbf{w}_n$ is correlated over elements, the same procedure as for the state-space model can be incorporated to describe $\mathbf{w}_n$ as a product of a matrix and a zero mean Gaussian noise vector with diagonal covariance matrix.

The factor graph can then be constructed as follows. The vector vertices are split into scalar vertices each of which representing the corresponding vector element. The

FG of the state-space model, see Figure 6.4, is then constructed based on substituting (6.19), (6.20) into (6.18) resulting in

$$p\left(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}\right) = \int \dots \int \prod_{i=1}^{p} \delta\left(\theta_{n,i} - \sum_{j=1}^{p} a_{i,j}\theta_{n-1,j} - \sum_{j=1}^{r} b_{i,j}u_j\right) \cdot \prod_{i=1}^{r} \mathcal{N}_{u_i}\left(0,\, \sigma_{u_i}^2\right) \mathrm{d}u_1 \dots \mathrm{d}u_r. \tag{6.28}$$

Similarly, the FG representing the likelihood function, see Figure 6.5, is constructed by substituting (6.26), (6.27) into (6.25) resulting in

$$p\left(\mathbf{x}_n|\boldsymbol{\theta}_n\right) = \int \dots \int \prod_{i=1}^{M} \delta\left(x_{n,i} - \sum_{j=1}^{M} h_{i,j}\theta_{n,j} - w_i\right) \mathcal{N}_{w_i}\left(0,\, \sigma_{w_i}^2\right) \mathrm{d}w_1 \dots \mathrm{d}w_M. \tag{6.29}$$

The update rules for the factor and variable nodes are depicted in Figure 6.6. Note that these are just special cases of those for the vector vertices in Figure 6.2, respecting more inputs. To derive the update rules for multiple input nodes, we resorted to a recursive evaluation of the update rules for a pair of input nodes.

The advantage of the scalar Kalman filter is that the update rules operate on scalars and simple arithmetic operations unlike the vector case. The complexity of the algorithm is quadratic in the number of states and linear in the number of observations. This is not true for the vector KF where the complexity is cubic in states and cubic or quadratic in observations depending on the form of the KF. However, the sum-product algorithm in the scalar case does not yield truly the MMSE estimates of the parameter $\boldsymbol{\theta}_n$, since the cycles are present in the graph. The complexity then will be a multiple of the number of iterations and the accuracy with convergence will have to be investigated for every single implementation of the scalar Kalman filter.

The *message passing algorithm* starts at variable nodes $\theta_{0,1},\dots,\ \theta_{0,p}$ where the messages sent to factor nodes $P_{1,1},\dots,\ P_{1,p}$ comprise the mean and variance of the [2]initial distribution $\lambda_{\theta_{0,i}\to P_{1,j}}\left(\theta_{0,i}\right) = \left\{\mu_{0,i},\, \sigma_{0,i}^2\right\}$ for $i,\, j \in \{1,\dots,\, p\}$ where $\mu_{0,i} = [\boldsymbol{\mu}_0]_i$ and $\sigma_{0,i}^2 = [\mathbf{C}_0]_{i,i}$. The messages from variable nodes $u_{0,i}$ are also sent to the factor nodes $P_{1,1},\dots,\ P_{1,p}$ $\lambda_{u_{1,i}\to P_{1,j}}\left(u_{1,i}\right) = \left\{0,\, \sigma_{u,i}^2\right\}$ and updates are calculated according to Figure 6.6. The branches to variable nodes $\theta_{1,1},\dots,\theta_{1,p}$ are disconnected and the resulting messages are sent back towards the variable nodes $\theta_{0,1},\dots,\ \theta_{0,p}$ and

[2]The initial cross-correlation between the parameter is not assumed. To regard it, one can resort to SVD decomposition.

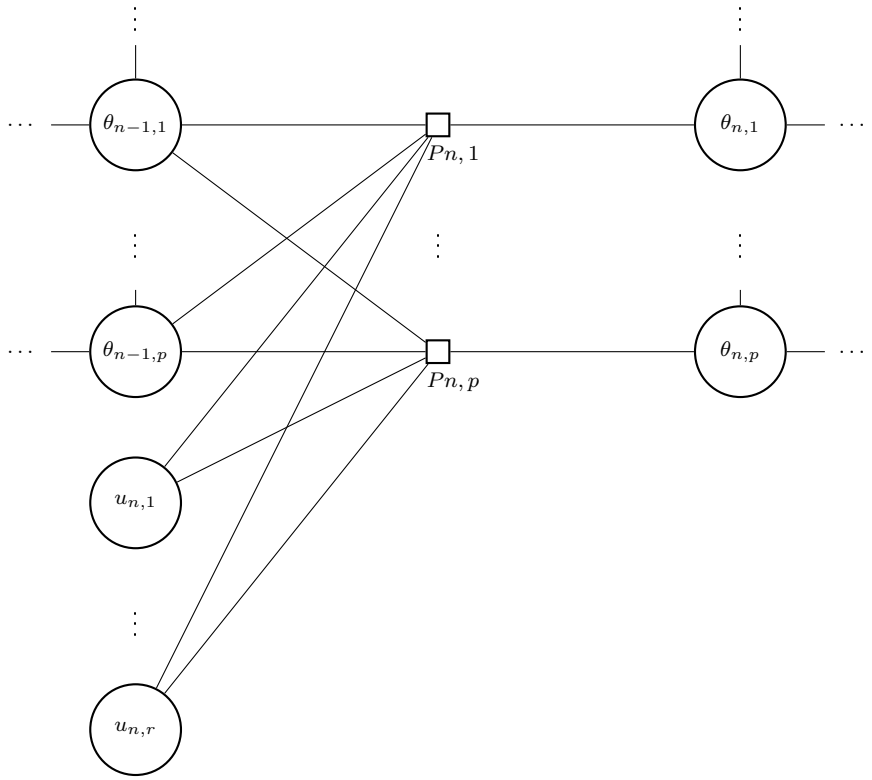


Figure 6.4: FG for state-space model of scalar KF

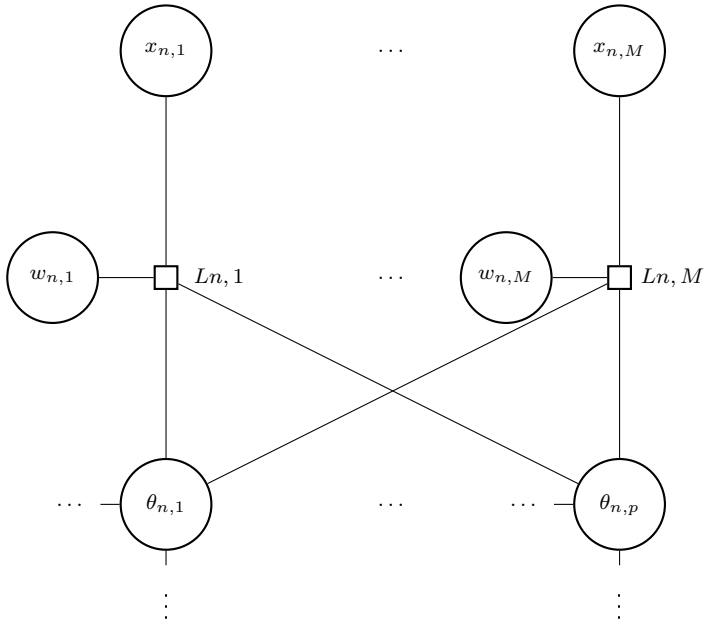
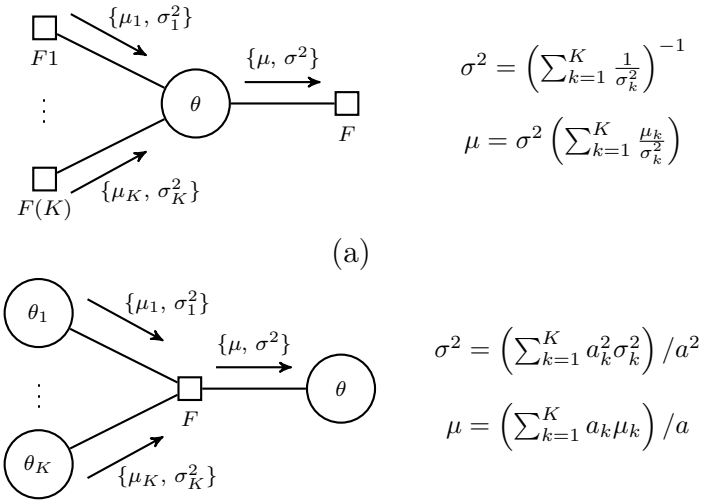


Figure 6.5: FG for likelihood function of scalar KF



(a)

$$p_F\left(\theta|\theta_1, \ldots, \theta_K\right) = \delta\left(a\theta - \sum_{k=1}^{K} a_k\theta_k\right)$$

(b)

Figure 6.6: Update rules for scalar vertices of KF - (a) variable node, (b) factor node

the iterations start. When the last iteration, the branches to the variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ are connected, the messages are sent to them. The observed values are sent to the likelihood factor nodes $\lambda_{x_{n,i} \to L_{n,i}}(x_{n,i}) = \{x_{n,i}, 0\}$ for $i \in \{1, \ldots, M\}$ and so do the noise variable nodes $\lambda_{w_{n,i} \to L_{n,i}}(w_{n,i}) = \{0, \sigma_i^2\}$. Next, the messages from the previous state-space factor nodes $P_{1,1}, \ldots, P_{1,p}$ are sent through variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ to the likelihood factor nodes $L_{1,1}, \ldots, L_{1,p}$ and the updates are therein calculated. The results are then sent back to variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ and then updated taking the unchanged messages from factor nodes $P_{0,1}, \ldots, P_{0,p}$. The branches to the future factor nodes $P_{2,1}, \ldots, P_{2,p}$ remain disconnected. The updated messages are sent towards factor nodes $L_{1,1}, \ldots, L_{1,p}$ and iterations are started in this way. The iterations are finished after the last updates at variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ for which the future variable nodes $P_{2,1}, \ldots, P_{2,p}$ are now connected. The messages sent to these factor nodes represent the beliefs and the means can be used to obtain the approximated MMSE estimates. And so continuous the algorithm sequentially.

The *number of flops* required to calculate one recursion is

$$\begin{aligned} \mathcal{O}_{\text{sKF}}\left(p,\, M,\, r,\, N_{\text{Iter., Ss.}},\, N_{\text{Iter., Lh.}}\right) &= N_{\text{Iter., Ss.}}\left(5p^2 + 5pr + p\right) \\ &+ \left(N_{\text{Iter., Ss.}} - 1\right)\left(4p^2 + 4pr\right) \\ &+ N_{\text{Iter., Lh.}}\left(9pM + 6M\right) \end{aligned} \tag{6.30}$$

where p is the size of the state-space vector $\boldsymbol{\theta}_n$, M is the size of the observation vector $\mathbf{x}_n$, r is the size of the driving noise vector $\mathbf{u}_n$, $N_{\text{Iter., Lh.}}$ is the number of the iterations among the likelihood factor nodes and the state variable nodes, $N_{\text{Iter., Lh.}}$ is the number of the iterations among the nodes.

6.4 Proposed Scalar Iterative Extended Kalman Filter on the FG

The *extended Kalman filter* can be also approximated on the scalar FG. Suppose that

$$\mathbf{a}\left(\boldsymbol{\theta}_n\right) = \begin{bmatrix} a_1\left(\theta_{n,1}, \ldots, \theta_{n,p}\right) \\ \vdots \\ a_p\left(\theta_{n,1}, \ldots, \theta_{n,p}\right) \end{bmatrix} \tag{6.31}$$

and

$$\mathbf{h}\left(\boldsymbol{\theta}_n\right) = \begin{bmatrix} h_1\left(\theta_{n,1}, \ldots, \theta_{n,p}\right) \\ \vdots \\ h_M\left(\theta_{n,1}, \ldots, \theta_{n,p}\right) \end{bmatrix}. \tag{6.32}$$

Let us next assume that $h_{i,j} = [\mathbf{H}]_{i,j}$ and $a_{i,j} = [\mathbf{A}]_{i,j}$. The conditional PDFs $p(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}, \mathbf{u}_n)$, $p(\mathbf{x}_n|\boldsymbol{\theta}_n, \mathbf{w}_n)$ both can be approximated for the EKF as Gaussian

$$p(\boldsymbol{\theta}_n|\boldsymbol{\theta}_{n-1}, \mathbf{u}_n) = \prod_{i=1}^{p} \delta\left(\theta_{n,i} - a_i(\theta_{n-1,1}, \ldots, \theta_{n-1,p}) - \sum_{j=1}^{r} b_{i,j}u_j\right) \tag{6.33}$$

$$p(\mathbf{x}_n|\boldsymbol{\theta}_n, \mathbf{w}_n) = \prod_{i=1}^{M} \delta(x_{n,i} - h_i(\theta_{n,1}, \ldots, \theta_{n,p}) - w_i). \tag{6.34}$$

The update rules of the scalar variable node remain unchanged compared to the KF's. However, the mean of the factor node update will change and the variance will remain unchanged. Assume the situation in Figure 6.6(b). Next, assume the PDF (=factor function) $p_F(\theta|\theta_1, \ldots, \theta_K)$ will have the following form

$$\begin{aligned} p_F(\theta|\theta_1, \ldots, \theta_K) &= \delta(a\theta - f(\theta_1, \ldots, \theta_K)) && (6.35)\\ &\approx \delta\left(a\left(\theta - \breve{\theta}\right) - \sum_{k=1}^{K} a_k\left(\theta_k - \breve{\theta}_k\right)\right) && (6.36) \end{aligned}$$

where f is K-dimensional function and $\breve{\theta}$, $\breve{\theta}_1, \ldots, \breve{\theta}_K$ are constant linearizing points such that

$$\breve{\theta} = f\left(\breve{\theta}_1, \ldots, \breve{\theta}_K\right). \tag{6.37}$$

The updated mean will have the following form

$$\mu = \breve{\theta} + \left(\sum_{k=1}^{K} a_k\left(\mu_k - \breve{\theta}_k\right)\right)/a. \tag{6.38}$$

In the scalar EKF, the linearization will generally take place in both state-space and likelihood factor nodes. The linearizing points for the state-space model nodes will be the estimated parameters from the previous time $\hat{\theta}_{n-1,i}$ where $i \in \{1, \ldots, p\}$. The linearizing points for the likelihood nodes will be the means of the output messages from the state-space model nodes. This complies with the vector EKF where predictions are used to linearize the nonlinear observation function $\mathbf{h}$.

The complexity of the scalar EKF is slightly higher than that of the scalar KF due to the necessity of evaluation of the Jacobians $\mathbf{A}$, $\mathbf{H}$ if they depend on time. The number of flops might be increased if functions $\mathbf{a}$, $\mathbf{h}$ are complicated. The linearizing point can be updated after every iteration to improve the convergence which requires the evaluations of the Jacobians.

Chapter 7

Overview of PVT Estimation/Filtering Algorithms

In this chapter, we derive the commonly used PVT estimation/filtering algorithms based on the general formulae from Chapters 3, 4, 6. These include the LS, WLS, EKF, UKF, and PF algorithms. In addition to it, the CRLB is derived for both the classical estimator and the Bayesian filter. Strictly speaking, the LS and WLS are of iterative nature and only approximate such estimates. Other classical estimation approaches such as MVU and ML are not discussed. The MVU estimator is difficult to derive due to a complicated form of the likelihood PDF. If considering Gaussian uncorrelated pseudorange and pseudorange measurements, as for the EKF, the WLS algorithm becomes the ML algorithm. We resort to the LS approach respecting conventions.

Finally, we incorporate the proposed algorithms of the scalar iterative EKF into the PVT problem. We show that by disconnecting the factor nodes representing the state-space model, a FG approximating the WLS algorithm can be obtained.

In the following text, we concatenate the vectors of pseudoranges $\boldsymbol{\rho}$ and pseudorange rates $\dot{\boldsymbol{\rho}}$, their observation functions $\mathbf{g}_{\mathbf{x}}$ and $\mathbf{g}_{\mathbf{v}}$, and the noise vectors $\mathbf{w}_{\boldsymbol{\rho}}$ and $\mathbf{w}_{\dot{\boldsymbol{\rho}}}$

$$\boldsymbol{\xi} \triangleq \begin{bmatrix} \boldsymbol{\rho} \\ \dot{\boldsymbol{\rho}} \end{bmatrix}, \quad \mathbf{h}(\boldsymbol{\gamma}) \triangleq \begin{bmatrix} \mathbf{g}_{\mathbf{x}}(\mathbf{x}_U, b) \\ \mathbf{g}_{\mathbf{v}}\left(\mathbf{v}_U, \dot{b}\right) \end{bmatrix}, \quad \mathbf{w} \triangleq \begin{bmatrix} \mathbf{w}_{\boldsymbol{\rho}} \\ \mathbf{w}_{\dot{\boldsymbol{\rho}}} \end{bmatrix}. \tag{7.1}$$

The covariance of the noise vector is denoted as $\mathbf{C}$. In case of the filtering methods, time index is associated with the corresponding quantities.

7.1 Classical Estimators

7.1.1 Cramer-Rao Lower Bound

Let us assume the measurement equation in the following form

$$\boldsymbol{\xi} = \mathbf{h}(\boldsymbol{\gamma}) + \mathbf{w} \tag{7.2}$$

where the likelihood function of the observations is Gaussian

$$p(\boldsymbol{\xi}|\boldsymbol{\gamma}) = \mathcal{N}_{\boldsymbol{\xi}}(\mathbf{h}(\boldsymbol{\gamma}), \mathbf{C}).$$

The elements of Fisher information matrix of the CRLB can be obtained using (3.27) as

$$\begin{aligned}
[\mathbf{J}(\boldsymbol{\gamma})]_{ij} &= \left[\frac{\partial \mathbf{h}(\boldsymbol{\gamma})}{\partial \gamma_i}\right]^T \mathbf{C}^{-1} \left[\frac{\partial \mathbf{h}(\boldsymbol{\gamma})}{\partial \gamma_j}\right] \\
&+ \frac{1}{2}\mathrm{tr}\left[\mathbf{C}^{-1}\frac{\mathbf{C}}{\partial \gamma_i}\mathbf{C}^{-1}\frac{\mathbf{C}}{\partial \gamma_j}\right] \qquad (7.3) \\
&= \left[\frac{\partial \mathbf{h}(\boldsymbol{\gamma})}{\partial \gamma_i}\right]^T \mathbf{C}^{-1} \left[\frac{\partial \mathbf{h}(\boldsymbol{\gamma})}{\partial \gamma_j}\right]. \qquad (7.4)
\end{aligned}$$

by extension to the matrix case, we get that

$$\mathbf{J}(\boldsymbol{\gamma}) = \mathbf{H}^T(\boldsymbol{\gamma})\mathbf{C}^{-1}\mathbf{H}(\boldsymbol{\gamma}). \tag{7.5}$$

If further $\mathbf{C}$ is assumed to be diagonal $\mathbf{C} = \mathrm{diag}\left(\sigma^2_{\rho,1}, \ldots, \sigma^2_{\rho,I}, \sigma^2_{\dot{\rho},1}, \ldots, \sigma^2_{\dot{\rho},I}\right)$, the evaluation of (7.4) would not require the matrix inversion. It should be noted that the CRLB of $\boldsymbol{\gamma}$ depends on the direction cosines.

7.1.2 Least Squares

Since the signal model for PVT estimation is nonlinear, see (1.30), the estimator must be iterative. Typically, the Gauss-Newton iterations are adopted due to simple implementation and fast convergence after the initialization discussed in Section 1.1.6.4. Let us denote the initial estimate as $\hat{\boldsymbol{\gamma}}_0$ and kth estimate as $\hat{\boldsymbol{\gamma}}_k$. The estimate is then obtained recursively, for $k = 1, \ldots, K$ where K is the number of iterations, as

$$\hat{\boldsymbol{\gamma}}_k = \hat{\boldsymbol{\gamma}}_{k-1} + \left(\mathbf{H}_k^T\mathbf{H}_k\right)^{-1}\mathbf{H}_k^T\left(\boldsymbol{\xi} - \mathbf{h}\left(\hat{\boldsymbol{\gamma}}_{k-1}\right)\right) \tag{7.6}$$

where $\mathbf{H}_k$ is $2I \times 8$ matrix such that

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}(\boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}}\right|_{\hat{\boldsymbol{\gamma}}_{k-1}}. \tag{7.7}$$

It can be easily shown that

$$\mathbf{H}_k = \begin{bmatrix} \mathbf{G}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_k \end{bmatrix} \tag{7.8}$$

where $\mathbf{G}_k$ is $I \times 4$ matrix called as *geometry matrix*

$$\mathbf{G}_k = \begin{bmatrix} -\mathbf{1}_1^T(\hat{\mathbf{x}}_{U,k-1}) & 1 \\ \vdots & \vdots \\ -\mathbf{1}_I^T(\hat{\mathbf{x}}_{U,k-1}) & 1 \end{bmatrix} \tag{7.9}$$

since $\mathbf{1}_i^T(\hat{\mathbf{x}}_{U,k-1})$ is the direction cosine vector between the user at position $\hat{\mathbf{x}}_{U,k-1}$ and the ith SV calculated according to (1.16). Note that the measurement equation is nonlinear for position and linear for velocity. In addition to it, matrix $\mathbf{H}_k$ is of the form that the velocity measurements do not support position estimates. Hence, we first iteratively estimate the user position and clock bias for $k = 1, \ldots, K$

$$\begin{bmatrix} \hat{\mathbf{x}}_{U,k} \\ \hat{b}_k \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_{U,k-1} \\ \hat{b}_{k-1} \end{bmatrix} + \left(\mathbf{G}_k^T\mathbf{G}_k\right)^{-1}\mathbf{G}_k^T\left(\boldsymbol{\rho} - \mathbf{g}_{\mathbf{x}}\left(\begin{bmatrix} \hat{\mathbf{x}}_{U,k-1} \\ \hat{b}_{k-1} \end{bmatrix}\right)\right) \tag{7.10}$$

and the user velocity and clock drift after the last iteration

$$\begin{bmatrix} \hat{\mathbf{v}}_U \\ \hat{\dot{b}} \end{bmatrix} = \left(\mathbf{G}_K^T\mathbf{G}_K\right)^{-1}\mathbf{G}_K^T\left(\dot{\boldsymbol{\rho}} - \begin{bmatrix} -\mathbf{1}_1^T(\hat{\mathbf{x}}_{U,K})\cdot\mathbf{v}_{S,1} \\ \vdots \\ -\mathbf{1}_I^T(\hat{\mathbf{x}}_{U,K})\cdot\mathbf{v}_{S,I} \end{bmatrix}\right). \tag{7.11}$$

7.1.3 Weighted Least Squares

To regard for signals with different quality, weighting can be incorporated to the least squares. The weighting matrix is typically constructed either based on the estimated C/N_0 values or based on the elevation angles as discussed in Section 1.1.6.5. Let $\mathbf{W}_{\mathbf{x}}$ be the weight matrix for current the pseudorange measurements, and let $\mathbf{W}_{\mathbf{v}}$ be the weight matrix for the current pseudorange rate measurements, the weighted least squares estimates of the user position and clock bias are for $k = 1, \ldots, K$

$$\begin{bmatrix} \hat{\mathbf{x}}_{U,k} \\ \hat{b}_k \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_{U,k-1} \\ \hat{b}_{k-1} \end{bmatrix} + \left(\mathbf{G}_k^T\mathbf{W}_{\mathbf{x}}\mathbf{G}_k\right)^{-1}\mathbf{G}_k^T\mathbf{W}_{\mathbf{x}}\left(\boldsymbol{\rho} - \mathbf{g}_{\mathbf{x}}\left(\begin{bmatrix} \hat{\mathbf{x}}_{U,k-1} \\ \hat{b}_{k-1} \end{bmatrix}\right)\right) \tag{7.12}$$

and the estimates of the user velocity and the clock drift are

$$\begin{bmatrix} \hat{\mathbf{v}}_U \\ \hat{\dot{b}} \end{bmatrix} = \left(\mathbf{G}_K^T \mathbf{W}_\mathbf{v} \mathbf{G}_K\right)^{-1} \mathbf{G}_K^T \mathbf{W}_\mathbf{v} \left(\dot{\boldsymbol{\rho}} - \begin{bmatrix} -\mathbf{1}_1^T \left(\hat{\mathbf{x}}_{U,K}\right) \cdot \mathbf{v}_{S,1} \\ \vdots \\ -\mathbf{1}_I^T \left(\hat{\mathbf{x}}_{U,K}\right) \cdot \mathbf{v}_{S,I} \end{bmatrix} \right). \tag{7.13}$$

7.1.4 Proposed Scalar Iterative Weighted Least Squares

The algorithm derived for scalar iterative extended Kalman filter using FG in Section 6.4 can be simplified to exclude the state-space model by disconnecting the state variable nodes across time. The estimation is then based only on the iterations on the FG in Figure 6.5.

7.2 Bayesian Estimators

7.2.1 Motion Models

The state-space model for a moving object with a stand-alone GNSS receiver has typically one of the three forms - position, velocity, and acceleration Gauss-Markov motion models. In all the cases, the driving noise vector $\mathbf{u}$ has diagonal covariance matrix $\mathbf{Q}$ constant in time, and matrix $\mathbf{B}$ transforming the driving noise vector from the *east-north-up* (ENU) coordinate frame into the ECEF coordinate frame. This allows us to model the motion with respect to the local coordinate frame. Typically, z user position coordinate changes less in time than x, y coordinates. However, to keep our approach simple we will model the motion directly in the ECEF coordinate frame letting $\mathbf{B}$ be the identity matrix.

The *position motion model* is intended for situations where the user moves slowly or does not move at all. It does not provide velocity estimates. The user PVT vector simplifies to

$$\boldsymbol{\gamma} = \begin{bmatrix} \mathbf{x}_U \\ b \end{bmatrix}. \tag{7.14}$$

The state-transition matrix is just the identity matrix

$$\mathbf{A} = \mathbf{I}_{4\times 4} \tag{7.15}$$

and the driving noise covariance matrix has four elements modeling the perturbation in position coordinates and the clock bias

$$\mathbf{Q} = \operatorname{diag}\left(\sigma_{u,x}^2, \ldots, \sigma_{u,b}^2\right). \tag{7.16}$$

The observations contain just the pseudoranges

$$\boldsymbol{\xi} = \boldsymbol{\rho}. \tag{7.17}$$

The measurement equation has the following nonlinear function projecting the true user position and clock bias into the noiseless pseudoranges

$$\mathbf{h}\left(\boldsymbol{\gamma}\right) = \mathbf{g}_{\mathbf{x}}\left(\mathbf{x}_U,\, b\right). \tag{7.18}$$

The noise vector then simplifies to $\mathbf{w} = \mathbf{w}_{\boldsymbol{\rho}}$.

In most of the commercial GNSS receiver, the *velocity motion model* is implemented. We inherently use the notation to account for such situation throughout the text. The PVT estimation vector is 8×1 vector containing the user position coordinates, clock bias, velocity coordinates, and clock drift.

$$\boldsymbol{\gamma} = \begin{bmatrix} \mathbf{x}_U \\ b \\ \mathbf{v}_U \\ \dot{b} \end{bmatrix}. \tag{7.19}$$

The state-transition matrix models the relation between the position and its derivative - velocity

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_{4\times4} & T_N \cdot \mathbf{I}_{4\times4} \\ \mathbf{0}_{4\times4} & \mathbf{I}_{4\times4} \end{bmatrix} \tag{7.20}$$

denoting the elapsed time between two successive PVT estimation updates with T_N. The perturbation is modeled only within the first derivative components

$$\mathbf{Q} = \mathrm{diag}\left(0, \ldots, 0,\, \sigma^2_{u,\dot{x}}, \ldots,\, \sigma^2_{u,\dot{b}}\right). \tag{7.21}$$

The measurement vector comprises the pseudorange vector and the pseudorange rate vector

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\rho} \\ \dot{\boldsymbol{\rho}} \end{bmatrix}. \tag{7.22}$$

The measurement function complicates to

$$\mathbf{h}\left(\boldsymbol{\gamma}\right) = \begin{bmatrix} \mathbf{g}_{\mathbf{x}}\left(\mathbf{x}_U,\, b\right) \\ \mathbf{g}_{\mathbf{v}}\left(\mathbf{v}_U,\, \dot{b}\right) \end{bmatrix} \tag{7.23}$$

with noise vector

$$\mathbf{w} = \begin{bmatrix} \mathbf{w}_{\boldsymbol{\rho}} \\ \mathbf{w}_{\dot{\boldsymbol{\rho}}} \end{bmatrix}. \tag{7.24}$$

The *acceleration motion model* is typically used in airborne applications. The user PVT vector is the 12×1 vector

$$\boldsymbol{\gamma} = \begin{bmatrix} \mathbf{x}_U \\ b \\ \mathbf{v}_U \\ \dot{b} \\ \mathbf{a}_U \\ \ddot{b} \end{bmatrix} \tag{7.25}$$

where $\mathbf{a}_U$ is the user acceleration $\mathbf{a}_U = \ddot{\mathbf{x}}_U$. The state-transition matrix and the driving noise covariance matrix are then

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_{4\times4} & T_N \cdot \mathbf{I}_{4\times4} & T_N^2/2 \cdot \mathbf{I}_{4\times4} \\ \mathbf{0}_{4\times4} & \mathbf{I}_{4\times4} & T_N \cdot \mathbf{I}_{4\times4} \\ \mathbf{0}_{4\times4} & \mathbf{0}_{4\times4} & \mathbf{I}_{4\times4} \end{bmatrix} \tag{7.26}$$

$$\mathbf{Q} = \text{diag}\left(0, \ldots, 0, \, \sigma^2_{u,\ddot{x}}, \ldots, \, \sigma^2_{u,\ddot{b}}\right). \tag{7.27}$$

The measurement equation remains as for the velocity motion model.

7.2.2 Posterior Cramer-Rao Lower Bound

Let us assume that the motion model is the first order Gauss-Markov model

$$\boldsymbol{\gamma}_n = \mathbf{A}\boldsymbol{\gamma}_{n-1} + \mathbf{u}_n \tag{7.28}$$

with the measurement equation

$$\boldsymbol{\xi}_n \;\; = \;\; \mathbf{h}_n\left(\boldsymbol{\gamma}_n\right) + \mathbf{w}_n \tag{7.29}$$

where $\mathbf{w}_n$ is a zero mean WGN vector. Due to the fact that matrix $\mathbf{Q}_n$ is singular, the Fisher information matrix must be recursively calculated using the form based on matrix inversion lemma 3.154

$$\mathbf{J}_{n+1}^{-1} = \left(\mathbf{H}_{n+1}^T \mathbf{C}_{n+1}^{-1} \mathbf{H}_{n+1}\right)^{-1} \,||\, \left(\mathbf{Q} + \mathbf{A}\mathbf{J}_n^{-1}\mathbf{A}^T\right) \tag{7.30}$$

where we deduced that

$$\begin{aligned} \mathrm{E}_{\boldsymbol{\gamma}_{n+1}}\left[\mathbf{H}_{n+1}^T\left(\boldsymbol{\gamma}_{n+1}\right)\mathbf{C}_{n+1}^{-1}\mathbf{H}_{n+1}\left(\boldsymbol{\gamma}_{n+1}\right)\right] &= \mathbf{H}_{n+1}^T\left(\tilde{\boldsymbol{\gamma}}_{n+1}\right)\mathbf{C}_{n+1}^{-1}\mathbf{H}_{n+1}\left(\tilde{\boldsymbol{\gamma}}_{n+1}\right) & (7.31) \\ &= \mathbf{H}_{n+1}^T\mathbf{C}_{n+1}^{-1}\mathbf{H}_{n+1}. & (7.32) \end{aligned}$$

Operation $\mathbf{X} \,||\, \mathbf{Y}$ of square matrices $\mathbf{X}$ and $\mathbf{Y}$ of the dimension denotes an anology to calculation of parallel resistance in electrical circuits

$$\mathbf{X} \,||\, \mathbf{Y} = \mathbf{X} \left(\mathbf{X} + \mathbf{Y}\right)^{-1} \mathbf{Y}. \tag{7.33}$$

The sum $\mathbf{X}$+$\mathbf{Y}$ must be nonsingular. It is interesting to note that inverse of the Fisher information matrix is a parallel combination of CRLB for the classical estimator, see 7.5, and contribution of the prediction uncertainty of the previous estimate.

7.2.3 Extended Kalman Filter

The following equations summarize the recursion of the EKF for PVT estimation.

Prediction:

$$\tilde{\boldsymbol{\gamma}}_n = \mathbf{A}\hat{\boldsymbol{\gamma}}_{n-1}. \tag{7.34}$$

Minimum Prediction MSE Matrix:

$$\tilde{\mathbf{M}}_n = \mathbf{A}\mathbf{M}_{n-1}\mathbf{A}^T + \mathbf{Q}. \tag{7.35}$$

Kalman Gain Matrix:

$$\mathbf{K}_n = \tilde{\mathbf{M}}_n \mathbf{H}_n^T \left(\mathbf{C}_n + \mathbf{H}_n \tilde{\mathbf{M}}_n \mathbf{H}_n^T\right)^{-1}. \tag{7.36}$$

Correction

$$\hat{\boldsymbol{\gamma}}_n = \tilde{\boldsymbol{\gamma}}_n + \mathbf{K}_n \left(\boldsymbol{\xi}_n - \mathbf{h}_n \left(\tilde{\boldsymbol{\gamma}}_n\right)\right). \tag{7.37}$$

Minimum MSE Matrix

$$\mathbf{M}_n = \left(\mathbf{I} - \mathbf{K}_n \mathbf{H}_n\right) \tilde{\mathbf{M}}_n. \tag{7.38}$$

For a large number of the tracked SVs, the information form of the EKF would more efficient since the matrix $\mathbf{C}_n$ is diagonal. Furthermore, matrix $\mathbf{A}$ is very sparse and the evaluation of $\tilde{\boldsymbol{\gamma}}_n$, $\tilde{\mathbf{M}}_n$ would involve significantly less operations than for a general matrix $\mathbf{A}$. The decomposition of matrix $\mathbf{H}_n$ according to (7.8) into two zero $I \times 4$ matrices and two $I \times 4$ geometry matrices may also lower the overall number of flops of the algorithm.

7.2.4 Scalar Iterative Extended Kalman Filter

Using the procedures discussed in Section 6.4, we can deduce the FG for PVT filtering as in Figures 6.4, 6.5. Thanks to the sparse element representation of matrix $\mathbf{A}$, the

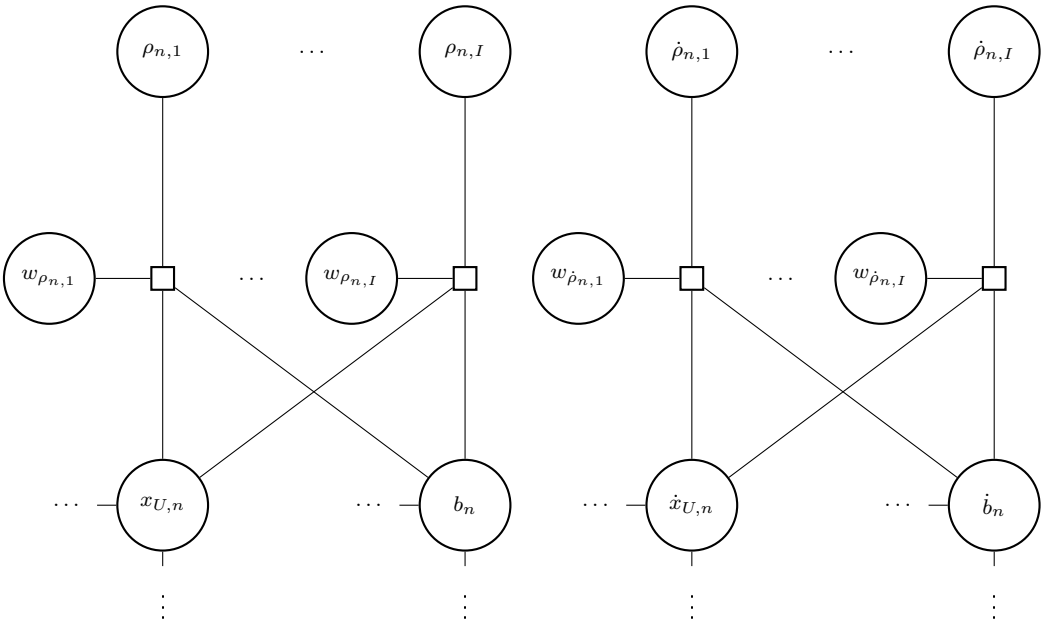


Figure 7.1: FG for the state-space model of the scalar EKF of PVT estimation.

state-space model FG will not contain cycles, see Figure 7.2, and the prediction can be accomplished in a single iteration. The likelihood FG, see Figure 7.1, does contain cycles and iterations will be needed to approximate the posterior distributions. The iteration will take place separately for the position and velocity parts of graph, as the likelihood function for the PVT vector $\boldsymbol{\gamma}$ can be decomposed as

$$p\left(\boldsymbol{\xi}|\boldsymbol{\gamma}\right) = p\left(\boldsymbol{\rho}|\mathbf{x}_U,\, b\right) p\left(\dot{\boldsymbol{\rho}}|\mathbf{v}_U,\, \dot{b}\right) \tag{7.39}$$

where

$$p\left(\boldsymbol{\rho}|\mathbf{x}_U,\, b\right) = \mathcal{N}\left(\mathbf{g}_{\mathbf{x}}\left(\mathbf{x}_U,\, b\right),\, \mathbf{C}_{\boldsymbol{\rho}}\right) \tag{7.40}$$

$$p\left(\dot{\boldsymbol{\rho}}|\mathbf{v}_U,\, \dot{b}\right) = \mathcal{N}\left(\mathbf{g}_{\mathbf{v}}\left(\mathbf{v}_U,\, \dot{b}\right),\, \mathbf{C}_{\dot{\boldsymbol{\rho}}}\right). \tag{7.41}$$

7.2.5 Unscented Kalman Filter

Assume that $\left\{\boldsymbol{\mathcal{Z}}^i_{n-1},\, \mathcal{W}^i_{n-1}\right\}_{i=0}^{2n_{\boldsymbol{\gamma}}}$, represent $2n_{\boldsymbol{\gamma}}+1$ sigma points and weights of $\boldsymbol{\gamma}_{n-1}$ with covariance matrix $\mathbf{M}_{n-1}$, respectively. The sigma points propagated through

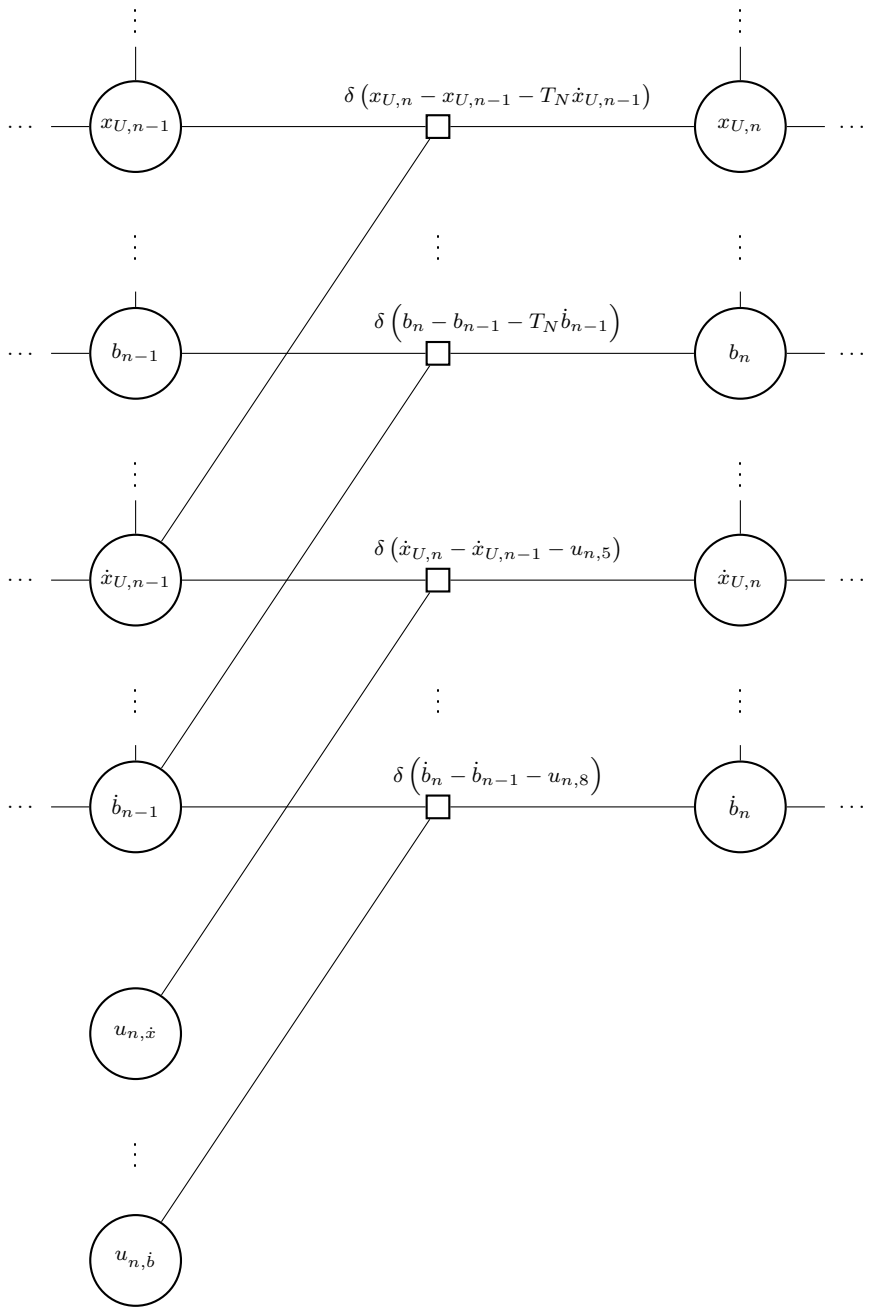


Figure 7.2: FG for the state-space model of the scalar EKF of PVT estimation.

the state-space model can be obtained as

$$\tilde{\boldsymbol{\mathcal{Z}}}_n \triangleq \begin{bmatrix} \tilde{\boldsymbol{\mathcal{Z}}}_n^0 \\ \vdots \\ \tilde{\boldsymbol{\mathcal{Z}}}_n^{2n_{\boldsymbol{\gamma}}+1} \end{bmatrix} = \mathbf{A}\tilde{\boldsymbol{\mathcal{Z}}}_{n-1}. \tag{7.42}$$

The prediction of the PVT vector $\tilde{\boldsymbol{\gamma}}_n$ and its covariance matrix $\tilde{\mathbf{M}}_n$ are then

$$\tilde{\boldsymbol{\gamma}}_n = \sum_{i=1}^{2n_{\boldsymbol{\gamma}}+1} \tilde{\boldsymbol{\mathcal{Z}}}_n^i \mathcal{W}_n^i \tag{7.43}$$

$$\tilde{\mathbf{M}}_n = \mathbf{Q} + \sum_{i=0}^{2n_{\boldsymbol{\gamma}}+1} \mathcal{W}_{n-1}^i \left(\tilde{\mathcal{Z}}_n^i - \tilde{\boldsymbol{\gamma}}_n\right)\left(\tilde{\mathcal{Z}}_n^i - \tilde{\boldsymbol{\gamma}}_n\right)^T. \tag{7.44}$$

To get the estimate of the PVT vector $\hat{\boldsymbol{\gamma}}_n$ and its covariance matrix $\mathbf{M}_n$, substitute $\tilde{\boldsymbol{\theta}}_n = \tilde{\boldsymbol{\gamma}}_n$, $\boldsymbol{\theta}_n = \boldsymbol{\gamma}_n$, $\mathbf{Q}_n = \mathbf{Q}$, $\mathbf{x}_n = \boldsymbol{\xi}_n$, $\mathbf{C}_{\mathbf{w}_n} = \mathbf{C}_n$ in recursion 4.22-4.28. After the recursion is finished, sigma points for the next step and its weights are generated using the UT on mean $\hat{\boldsymbol{\gamma}}_n$ and covariance matrix $\mathbf{M}_n$.

7.2.6 Particle Filter (Bootstrap Filter)

The implementation of the Bootstrap particle filter is straightforward. The samples are generated from the previous estimate $\hat{\boldsymbol{\gamma}}_{n-1}$ and its covariance $\mathbf{M}_{n-1}$, which are assumed to be Gaussian, according to 4.65

$$\boldsymbol{\gamma}_n^i \sim \mathbf{A} \cdot \left(\hat{\boldsymbol{\gamma}}_{n-1} + \mathbf{L}_{n-1}\boldsymbol{\eta}_n^i\right) \tag{7.45}$$

where $\boldsymbol{\eta}_n^i \sim \boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I}_{8\times 8})$ in case of the assumed velocity motion model, and $\mathbf{L}_{n-1}$ is a matrix such that $\mathbf{M}_{n-1} = \mathbf{L}_{n-1}\mathbf{L}_{n-1}^T$. The particle is assigned the weight by substitution into the likelihood function as in 4.66

$$\tilde{w}_n^i = \exp\left(-\frac{1}{2}\left(\boldsymbol{\xi}_n - \mathbf{h}_n\left(\boldsymbol{\gamma}_n^i\right)\right)^T \mathbf{C}_n^{-1}\left(\boldsymbol{\xi}_n - \mathbf{h}_n\left(\boldsymbol{\gamma}_n^i\right)\right)\right). \tag{7.46}$$

The particles are then normalized according to 4.67. The effective number of samples N_{eff} is calculated. If N_{eff} is less than $2N_s/3$, resampling from 4.1 is done. The final estimate is obtained according to 4.68, covariance matrix as for the UKF

$$\mathbf{M}_n = \sum_{i=0}^{N_s-1} w_n^i \left(\boldsymbol{\gamma}_n^i - \hat{\boldsymbol{\gamma}}_n\right)\left(\boldsymbol{\gamma}_n^i - \hat{\boldsymbol{\gamma}}_n\right)^T. \tag{7.47}$$

Samples $\boldsymbol{\gamma}_n^i$ are regenerated at every time step.

Chapter 8

Simulation and Experimental Results

In this chapter, we investigate the performance of the proposed suboptimal scalar iterative EKF using FG. This method is compared to the "standard" EKF. Nonlinear methods such as UKF, GF, PF are not included to facilitate the simulation and development time. The motivation for this is to build evidence that the tedious derivations throughout this study can meet practical implementations. We focus on two crucial characteristics - convergence, accuracy. In real life, the model of our parameter or our channel does not always reflect reality, therefore we also investigate the cases where the state-space model is designed for much lower dynamics than the user moves with.

In Figure 8.1, we plot the number of flops of the FG-based filter for the first three iterations and the number of flops of the EKF when implemented efficiently in the information form. Note that the number of flops is lower for the FG-based filter with a single iteration, for two iterations both curves are close to each other, and for three iterations the EKF has lower number of flops. During the simulation results, we will notice that a single iteration can offer similar accuracy to the EKF in low dynamic scenarios, but the performance improves with iterations as the dynamics increase and the EKF can be outperformed. Hence, the FG-based filter offers compromises in accuracy and complexity.

Firstly, we deliver a simple simulation where the convergence and accuracy in a scalar tracking architecture is investigated using Monte Carlo methods, being the subject of Section 8.1. In Section 8.2, the filters are incorporated into a vector tracking architecture. The former simulation assumes Gaussian uncorrelated measurements of pseudoranges and pseudorange rate, whereas the latter produces measurements that are correlated and biased by randomly generated values depending on the satellite elevations. Section 8.3 presents few case studies of the experiment

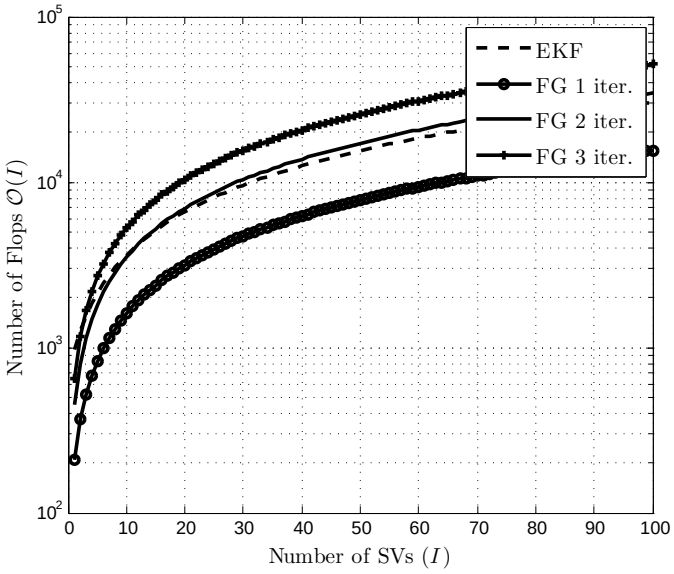


Figure 8.1: Complexity comparison of the PVT filtering algorithms. Symbol $\mathcal{O}(I)$ denotes the number of floating point operations (flops) depending on the number of SVs denoted as I

with a real-time receiver - WNav, working in the scalar tracking architecture and in the traditional concept.

8.1 Simulation - Scalar Tracking Architecture

In this simulation, we investigate convergence and accuracy of the EKF and the FG-based scalar iterative EKF algorithm. The model we adopt is basic. The pseudoranges and pseudorange rates are Gaussian and uncorrelated without any biases. The goal is to evaluate the preliminary performance characteristics in a simple manner and over a large number of realizations. In the first subsection, we introduce the simulation scenario, in the next two subsections the convergence and accuracy results are discussed and finally summarized in the last subsection.

8.1.1 Simulation Scenarios

We consider from 16 to 64 visible SVs randomly distributed over the open sky, scenarios with low dynamics ($a = 0.1\,\mathrm{m/s^2}$) and scenarios with moderate dynamics ($a = 1\,\mathrm{m/s^2}$). The user moves in a circle on the surface of the Earth with radius

1 km, velocity 10 m/s for low dynamics and 33 m/s for moderate dynamics. The estimates were updated every ten times per second $T_N = 0.1$ s.

The standard deviation of the generated pseudorange measurements was $\sigma_\rho = 1$ m, and $\sigma_{\dot{\rho}} = 0.05$ m/s of the pseudorange rate measurements. These quantities were generated as uncorrelated Gaussian random variables with mean equal to the noiseless values with respect to the geometry and clock offsets. The standard deviation of the elements of the driving noise vector was assumed constant - either $\sigma_u = 0.01$ m/s for high filtering or $\sigma_u = 0.1$ m/s for low filtering. We repeated each simulation 1000 times and simulated over 1000 s. The convergence and accuracy were examined for various number of iterations $N_{\text{Iter.}} \triangleq N_{\text{Iter. Lh.}}, N_{\text{Iter. Ss.}} = 1$.

To evaluate the performance characteristics, we define the total position error (TPE) of the position and clock bias estimation as

$$\text{TPE} = \frac{1}{NL}\sum_{n=1}^{N}\sum_{l=1}^{L}\sqrt{(\hat{x}_n^{(l)} - x_n^{(l)})^2 + (\hat{y}_n^{(l)} - y_n^{(l)})^2 + (\hat{z}_n^{(l)} - z_n^{(l)})^2 + (\hat{b}_n^{(l)} - b_n^{(l)})^2} \tag{8.1}$$

where n is the time index, N is the number of observations in time, l is the index of the realization, L is the number of realizations. Similarly, we define the total velocity error (TVE) as

$$\text{TVE} = \frac{1}{NL}\sum_{n=1}^{N}\sum_{l=1}^{L}\sqrt{(\hat{\dot{x}}_n^{(l)} - \dot{x}_n^{(l)})^2 + (\hat{\dot{y}}_n^{(l)} - \dot{y}_n^{(l)})^2 + (\hat{\dot{z}}_n^{(l)} - \dot{z}_n^{(l)})^2 + (\hat{\dot{b}}_n^{(l)} - \dot{b}_n^{(l)})^2}. \tag{8.2}$$

We claim that the divergence is reached if TPE>100 m or TVE>2 m/s.

8.1.2 Convergence

The convergence histograms of the EKF and FG-based scalar iterative EKF algorithms are depicted in Figure 8.2. In Figure 8.2(a), the driving noise std. equals the change of the velocity vector in magnitude over an epoch. In this case, we can infer that the convergence probability is strictly dependent on the number of visible SVs and is always worse for the FG-based EKF filter than for the standard EKF filter. In Figure 8.2(b), the same scenario is assumed for a larger number of SVs. For the number of visible SVs equal sixteen and more, no divergence has been observed in 1000 repetitions. This fact substantiates us to employ the iterative filtering for large data vectors. Luckily, we see that the convergence is relatively fast and can be reached within few iterations.

In the experiment in Figure 8.2(c), we increase the user velocity and produce the user acceleration of 1 m/s. The driving noise is left unchanged so that the velocity changes approximately ten times faster than the state-space model assumes. We observe that the FG-based filter and the EKF can still withstand such unmodeled dynamics and converge similarly as in the previous case. However, this is not the case in Figure 8.2(d) where the driving noise variance is increased to 0.1 m/s to account for the change in velocity. The lack of averaging of the iterative algorithm causes divergence even for a large number of SVs of the FG filter, the EKF remains stable. We deduce that if the dynamics increase significantly, we had better not model it. If the modeled dynamics is acceptably low at high dynamics, iterations generally improve convergence probability.

8.1.3 Accuracy

In Figure 8.4, we illustrate the accuracy of the algorithms. The TPEs are in the left column, whereas the TVEs are in the right column. Each row corresponds to the same scenario. In Figure 8.4(a-b), the same low dynamics matched model is considered. In this case, the FG-based filter outperforms the EKF if the number of SVs is larger than 16 and even for a single iteration. The explanation for this surprising fact is that the EKF is a suboptimal approximation of the KF based on first order Taylor linearization. The FG-based filter linearizes the variables in a scalar, but different, manner and the performance of both filters may generally differ depending on different factors.

If we increase the acceleration as in Figure 8.4(c-d), the FG-based filter outperforms the EKF in accuracy for larger number of SVs or for more than one iteration. In this case, the performance is improved over iterations due to the fact that a linearizing point is calculated for each iteration. The linearization fails for a large position difference between the predicted (also linearizing) value and the true value. Here, we iteratively shift towards the true value with iterations, whereas the EKF does this only once at an epoch.

If we model the higher dynamics and remain stable as in Figure 8.4(e-f), the EKF performs better and the iteration do not improve the accuracy, but the difference in accuracy between the two filters decays with larger number of the SVs.

8.1.4 Summary

To summarize the simulation results, we recall that one should use the FG-based filter for PVT estimation to lower the complexity of the algorithm only if the number

of visible SVs is larger than 16 to ensure the convergence. If the user is expected to move from time to time with higher dynamics, we had better not model it and rather increase the number of iterations. The accuracy can be improved in this case compared to the EKF.

8.2 Simulation - Vector Tracking Architecture

In this section, we demonstrate the functionality of the proposed algorithm in the vector tracking architecture. We present a case study where we compare the scalar tracking architecture using the EKF with the vector tracking architecture with the EKF or the FG-based scalar iterative EKF. This time correlation among the pseudo-ranges will be introduced and the biases in the measurement caused by atmospheric effects will be added. We demonstrate that the FG-based filter can withstand these anomalies even in the vector tracking architecture. The first subsection shortly explains the employed simulation methodology. In the second subsection, the simulation results are discussed and the last subsection highlights the key implementation aspects.

8.2.1 Simulation Methodology

The simulation setup is documented in Table 8.2. Two user scenarios were investigated: low dynamics (LD) scenario, and moderate dynamics (MD) scenario. In either scenario, the user moves in a circle with constant circular orbit speed. The TPE and TVE were calculated over time. For the first part of the simulation, we assume that the user is in an open-sky scenario, whereas in the second part C/N_0 of all visible SVs is swept from 50 dB-Hz down to 10 dB-Hz. Random atmospheric delay errors were artificially added to the propagation times with distribution depending on the elevation angle of the SV. Example values of these errors for the simulation in Figure 8.4(a,b) are in Table 8.1. Oscillator phase noise was added to the code delay and the carrier phase. Finite bandwidth of 10 MHz was introduced. The phase noise values $\mathcal{L}(f_0)$, $\mathcal{L}(f_1)$ at frequencies f_0, f_1, respectively, for the carrier phase are in Table 8.2. No multipath, interference, jamming and fading effects were examined.

The simulation complexity has been reduced by avoiding the bit-true multiplication and accumulation (MAC) between the received IF signal samples and the generated replicas. The signals at the output branches of the correlators were generated based on the approximate expressions representing such signals [37, 45, 63, 106].

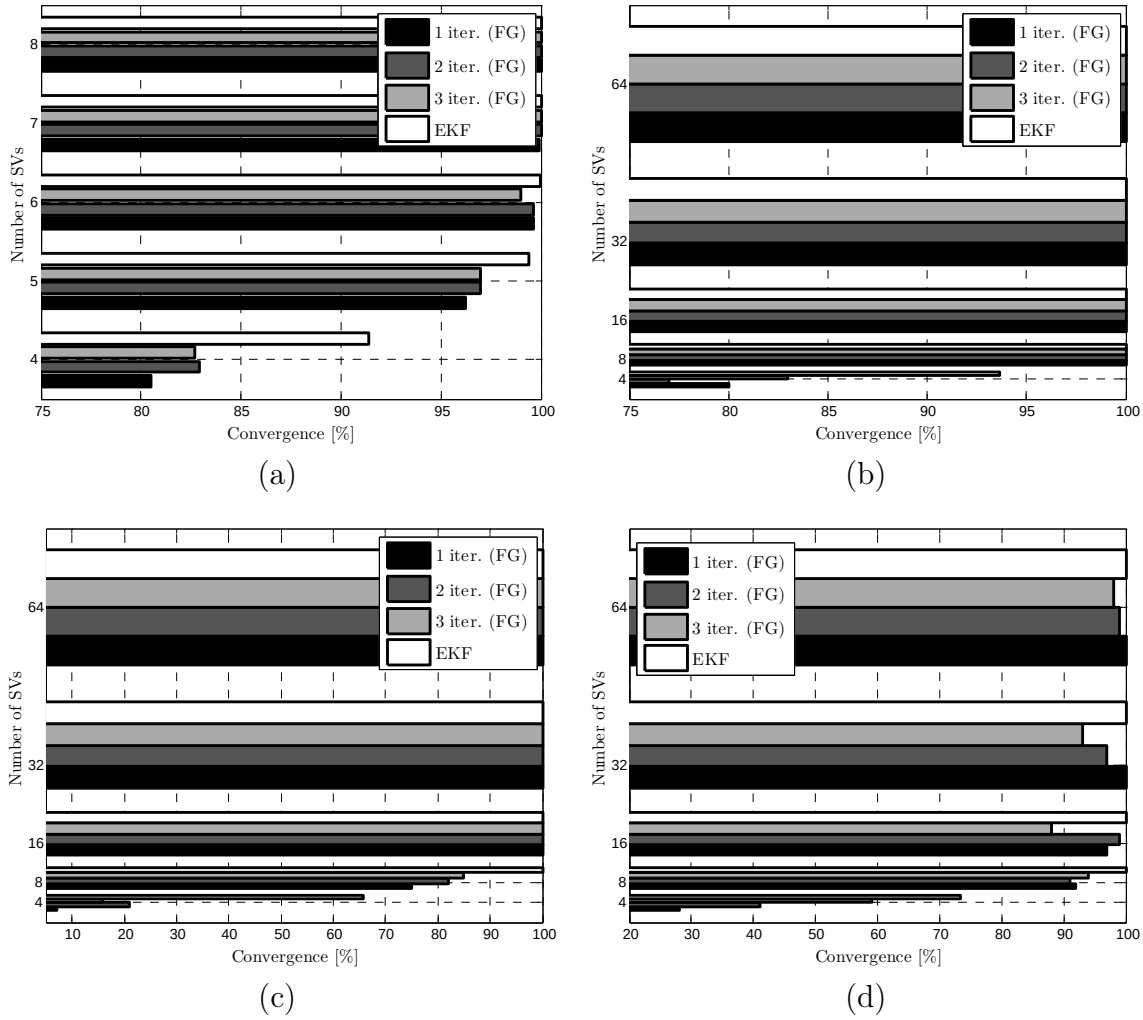


Figure 8.2: Simulation results - convergence. (a) $a = 0.1\,\mathrm{m/s^2}$, $\sigma_u = 0.01\,\mathrm{m/s}$ detail on low number of visible SVs, (b) $a = 0.1\,\mathrm{m/s^2}$, $\sigma_{u_u} = 0.01\,\mathrm{m/s}$, (c) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.01\,\mathrm{m/s}$, (d) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.1\,\mathrm{m/s}$.

Table 8.1: Examples of atmospheric errors (AE) with elevations of the SVs (El.), for simulation in Figures 8.4 (a, b)

El. [°]	32	42	9	10	21	6	33	40	6
AE [m]	1.1	-0.3	-3.7	0.9	-0.1	-2.3	1.3	-2.0	-2.5

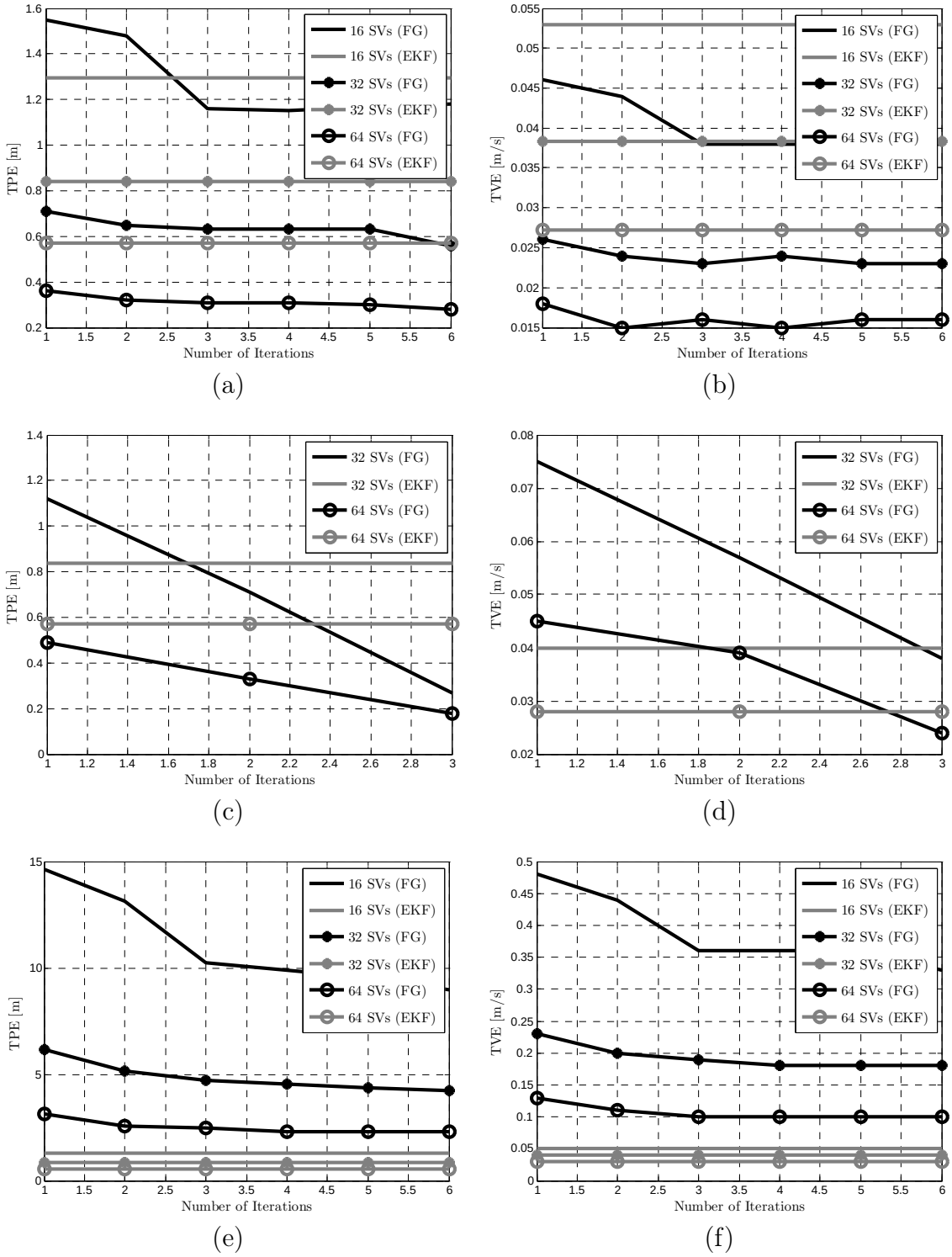


Figure 8.3: Simulation results - accuracy. Total position errors (TPE) are in the left column, total velocity errors (TVE) are in the right column. (a), (b) $a = 0.1\,\mathrm{m/s^2}$, $\sigma_u = 0.1\,\mathrm{m/s}$, (c), (d) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.01\,\mathrm{m/s}$, (e), (f) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.1\,\mathrm{m/s}$

Table 8.2: Simulation Setup

Testing Signal	GPS L1 C/A
Predetection Integration Time	$N_b N_c T_c = 20$ ms
Navigation Update Time	$T_N = 0.1$ s
IF Filter Bandwidth	$BW = 10$ MHz
Code Delay Detector	Normalized Power
Carrier Phase Detector	Atan2
C/N_0 Estimator	Squaring [81]
Prefilter	Sequential WLS
User Velocity, Radius, Accel. (LD)	2 m/s, 100 m, 0.04 G
Velocity Driv. Noise Std. (LD)	$\sigma_{\dot{x}} = \sigma_{\dot{y}} = \sigma_{\dot{z}} = 1$ m/s
Clk. Drift Driv. Noise Std. (LD)	$\sigma_{\dot{b}} = 10^{-2}$ m/s
User Velocity, Radius, Accel. (MD)	30 m/s, 100 m, 1 G
Velocity Driv. Noise Std. (MD)	$\sigma_{\dot{x}} = \sigma_{\dot{y}} = \sigma_{\dot{z}} = 20$ m/s
Clk. Drift Driv. Noise Std. (MD)	$\sigma_{\dot{b}} = 10^{-3}$ m/s
Number of Visible SVs	$I = 9$
Init. Lattitude, Longitude, Altitude	0°, 0°, 0 m
Phase Noise	$\mathcal{L}(1\,\text{Hz}) = -30$ dBc/Hz
	$\mathcal{L}(10\,\text{Hz}) = -50$ dBc/Hz
DLL (Order, Bandwidth)	1. order, $B_n = 0.1$ Hz
FLL (Order, Bandwidth)	2. order, $B_n = 10$ Hz

8.2.2 Results

The simulation results documenting the open-sky scenario are in Figures 8.4, 8.5. Figures 8.4(a-d) depict the situation for LD, whereas Figures 8.5(a-d) depict MD. TPE and TVE are plotted for a single iteration ($N_{\text{Iter.}} = 1$) and for three iterations ($N_{\text{Iter.}} = 3$). The proposed FG-based algorithm incorporated to the vector tracking architecture (FG-VDLL/FG-VFLL) is compared with the EKF of the scalar tracking architecture (DLL/FLL) and with the EKF of the vector tracking architecture (VDLL/VFLL). All algorithms adopt identical motion model. Second order FLL aids first order DLL in the scalar tracking loops.

It is clear from Figures 8.4(a,b), 8.5(a,b) that a single iteration on the FG results in comparable position and velocity filtering errors as for the EKF vector tracking loop in an open-sky scenario at low dynamics. An increased number of iterations slightly improves the performance at low dynamics, compare Figures 8.4(a,c) and Figures 8.4 (b,d). At moderate dynamics, both VDLL/VFLL and FG-VDLL/FG-VFLL perform similarly for $N_{\text{Iter.}} = 3$, see Figure 8.5(c,d). For $N_{\text{Iter.}} = 1$ conclusions about precision are difficult to make, see Figures 8.5(a,b). The reason is that our motion model does not regard user acceleration. However, the figures document that the proposed method might be able to withstand such unmodeled anomalies, likewise the EKF method. The improper choice of the motion model here causes that the scalar architecture outperforms the vector one for velocity estimation at moderate dynamics, see Figures 8.5(b,d). This could be claimed as a disadvantage of the vector tracking architecture.

In Figures 8.6(a,b), we sweep C/N_0 for all the tracked SVs, see Figure 8.6(c), and observe the stability of the proposed FG-based algorithm in comparison with the EKF-based one. Moderate dynamics and a single iteration are considered. It is apparent from the figures that the FG-VDLL/FG-VFLL loop looses lock at approximately 2 dB lower C/N_0 than the VDLL/VFLL loop. Fig. 8.6(d) depicts the C/N_0 estimate errors of the FG-VDLL/FG-VFLL channels.

8.2.3 Implementation Aspects

Likewise for the EKF architecture, it is clear that precision and stability of the loop is highly dependent on the choice of the motion model, predetection integration time and the navigation update time. When a proper motion model is selected, the performance can be improved with increased number of iterations on the FG. But with low number of iterations, the navigation processor might be able to operate at higher update rate which would foster the overall performance.

In our simulation, we did not model any delay between the SV channels and the

navigation processor. It represents the situation of a SDR receiver (ipexSR [15], WNav - only PC processing [16]). On the other, the performance might be worse for traditional receiver architectures (WNav - local tracking channels in FPGA, the navigation processor in PC [16]) where such a delay occurs.

8.3 Experiment - Scalar Tracking Architecture

In this section, we demonstrate the functionality of the proposed FG-based scalar iterative EKF algorithm in a scalar tracking architecture using a GNSS receiver developed at CTU in Prague [16] and high-fidelity GPS L1 Spirent simulator GSS6560. The first subsection shortly explains the employed methodology. In the second subsection, results are discussed.

8.3.1 Methodology

We assume that the user moves in a circle with constant circular orbit speed in an open-sky scenario. Firstly, we place the user to 14 static points on the Earth, see Figure 8.7, at a given time and investigate the filtering performance. Secondly, two dynamic scenarios ($a = 1\,\mathrm{m/s^2}$, $a = 10\,\mathrm{m/s^2}$) with radius 10 km, 1 km and velocity 100 m/s are supposed. The number of visible SVs has always been 11.

The measurements were taken for 1 hour. The navigation update time was $T_N = 0.1$ s. Second order DLL, PLL were used with equivalent loop noise bandwidth 0.5 Hz, 30.0 Hz, respectively. The driving noise std. was always 0.01 m/s which mismatches the motion model, but still filters out. The FG-based filter ran with three iterations.

8.3.2 Results

In Figure 8.8, we plot the results obtained by the WNav receiver. The left columns depict TPEs and the right column depict TVEs. Figures 8.8(a-b) are averaged versions of the TPEs and TVEs for the FG-based and the EKF filters applied to the static user. The accuracy of both filters is comparable. However, convergence was observed only in 10/14 cases. The reason is as discussed in Section 8.1 - low number of the SVs causes divergence.

In Figures 8.8(c-d), the user moves with acceleration $a = 1\ \mathrm{m/s^2}$, in Figures 8.8(e-f) with acceleration $a = 10\ \mathrm{m/s^2}$ which is not far from the model of the filter. The important fact is that the FG-based filter remains stable and its accuracy is comparable to the EKF's.

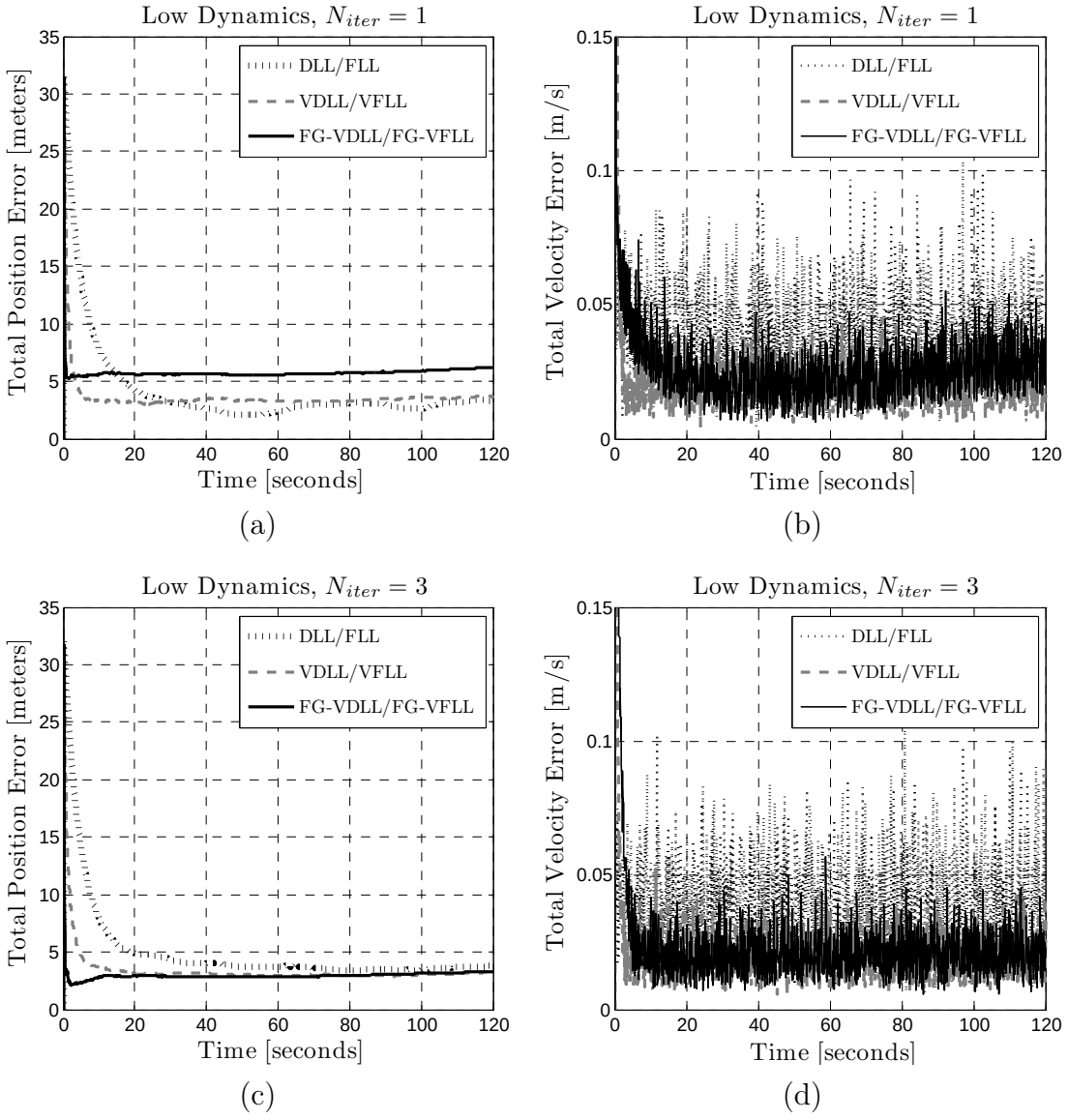


Figure 8.4: Vector tracking - simulation results - LD. (a) TPE, $N_{\text{Iter.}} = 1$, (b) TVE, $N_{\text{Iter.}} = 1$, (c) TPE, $N_{\text{Iter.}} = 3$, (d) TVE, $N_{\text{Iter.}} = 3$.

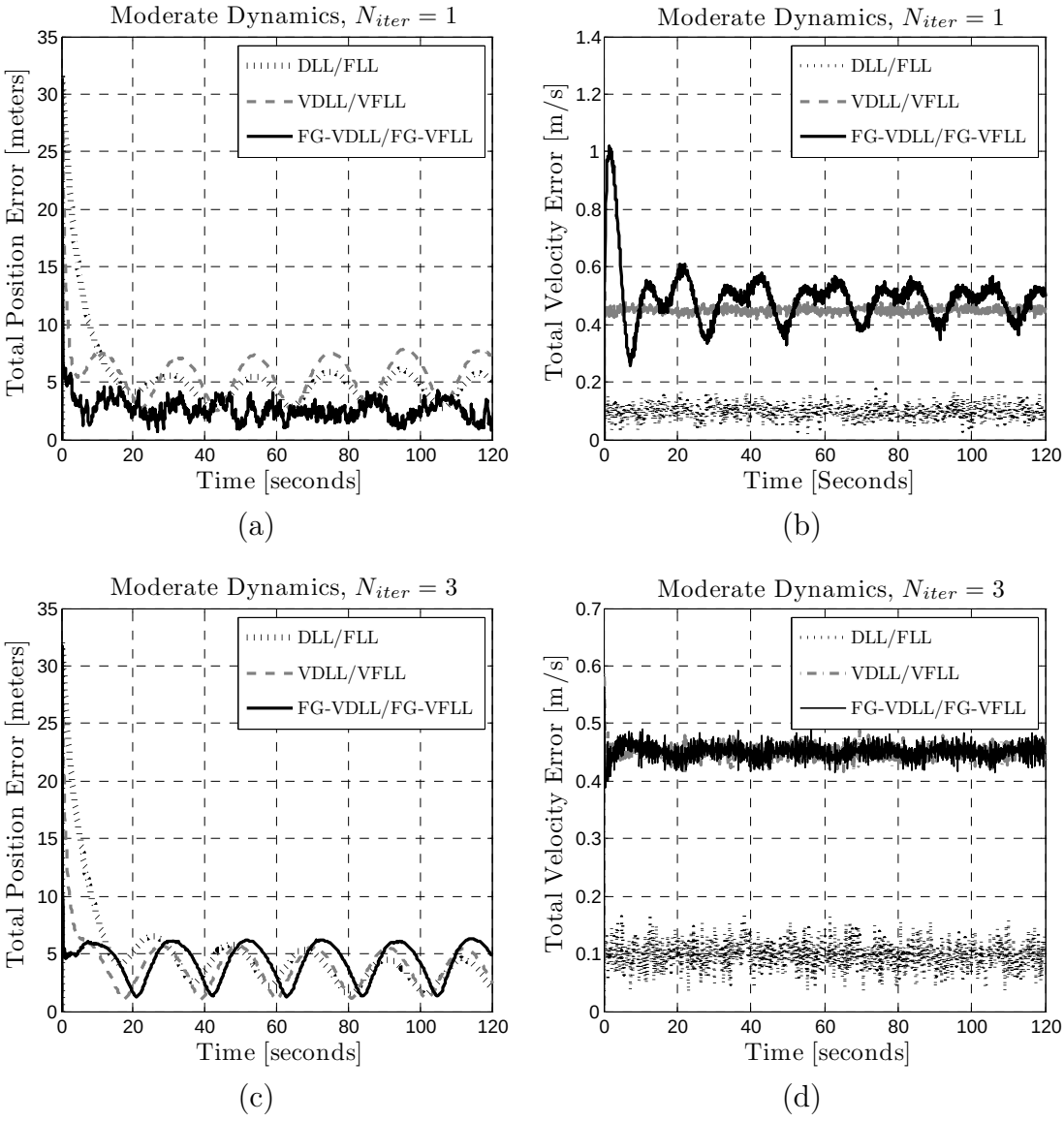


Figure 8.5: Vector tracking - simulation results - MD. (a) TPE, $N_{\text{Iter.}} = 1$, (b) TVE, $N_{\text{Iter.}} = 1$, (g) TPE, $N_{\text{Iter.}} = 3$, (h) TVE, $N_{\text{Iter.}} = 3$.

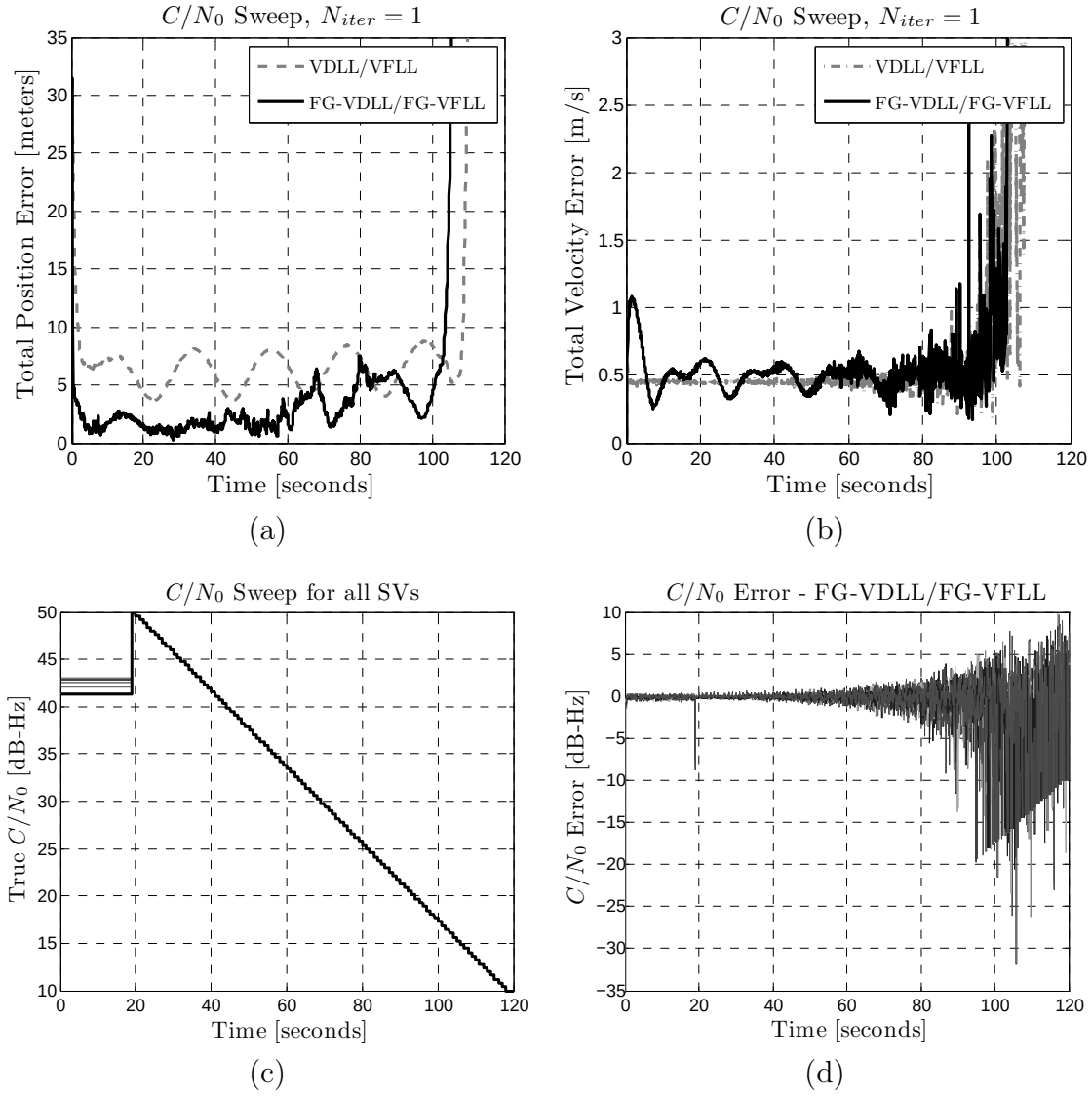


Figure 8.6: Vector tracking - simulation results - C/N_0 sweep. (a) TPE at C/N_0 sweep, (b) TVE at C/N_0 sweep, (c) swept C/N_0 of all visible SVs, (d) C/N_0 estimate error for all visible SVs. Figures (j-l) refer to MD, $N_{\text{Iter.}} = 1$.

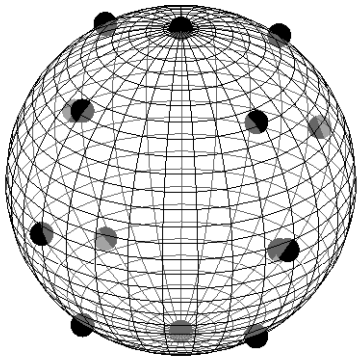


Figure 8.7: User positions on the Earth - static experiment (14 positions)

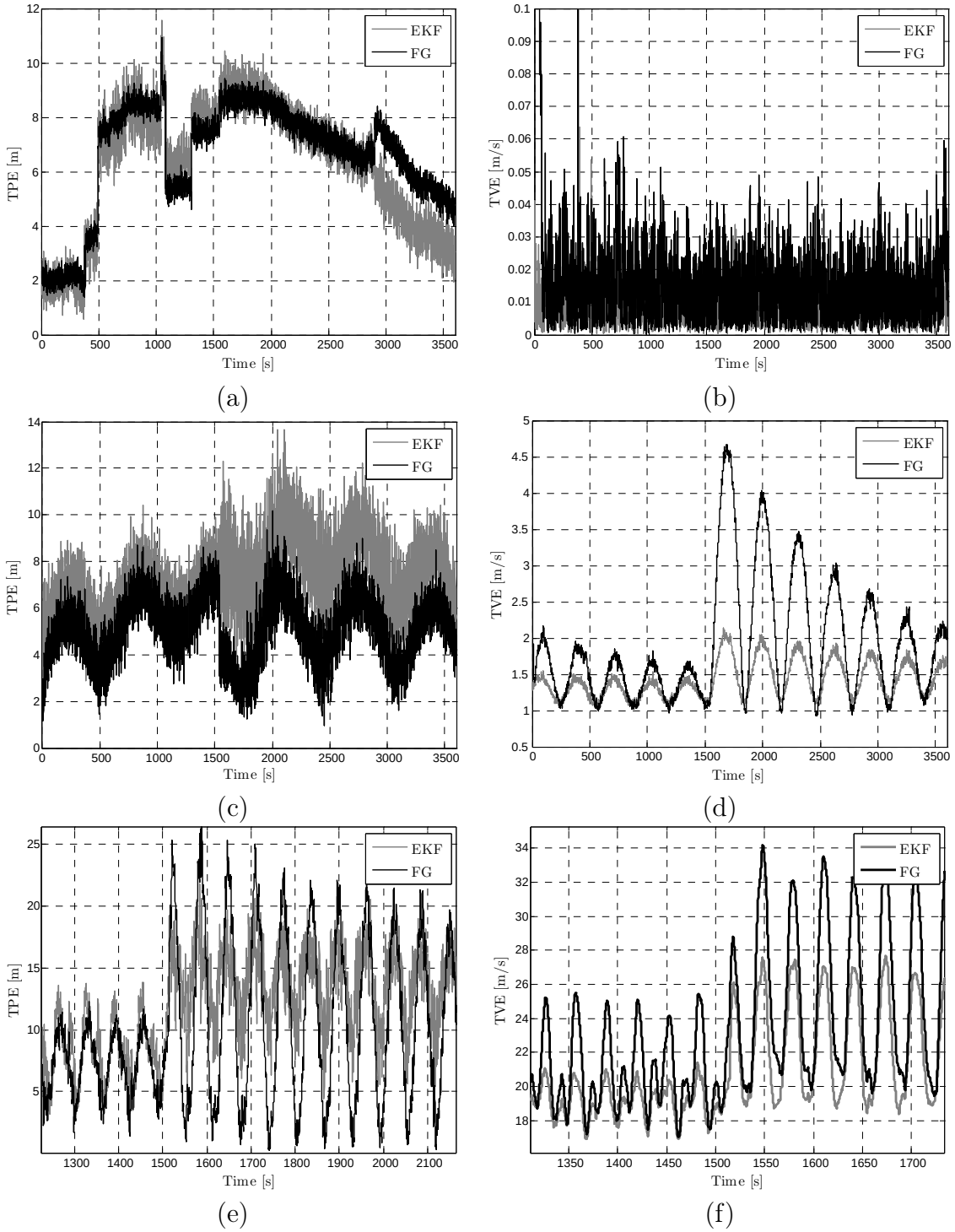


Figure 8.8: Experimental results - accuracy. Total position errors (TPE) are in the left column, total velocity errors (TVE) are in the right column. (a), (b) averaged errors of static scenarios, (c), (d) dynamic scenario $a = 1\,\mathrm{m/s^2}$, (e), (f) detail on dynamic scenario $a = 10\,\mathrm{m/s^2}$

Conclusion

In the thesis, we summarized basic aspects of the signal processing algorithms in GNSS. We formed measurement models of the PVT estimation for three receiver architectures - scalar tracking architecture, vector tracking architecture, and direct positioning architecture. The Witch Navigator was introduced as a testing platform for the study. The student is an active member of the team developing this receiver, and is responsible for the PVT estimation and its integration with the signal tracking algorithms.

In order to make this document comprehensive, we included the overview of classical and Bayesian estimation theory. Then, the topic of Bayesian filters was extended for "advanced" nonlinear filters such as UKF, GF, PF. The principles of system modeling with factor graphs and the generic sum-product algorithm were presented with a simple illustrative example. A factor graph model for a general Bayesian filter was then presented and the Kalman filter was derived for Gaussian PDF representation, the corresponding update rules and message passing algorithm.

Our novelty was introduced by first a simple linearization of the update rules and converting the algorithm to the EKF. Then, we split the vector nodes into scalar ones and derived a *general scalar iterative (extended) Kalman filter*. The algorithm involves only *scalar operations*, is *fully distributed*, hence suitable for parallel implementation. However, the *convergence must be verified for each particular implementation of this filter* using Monte Carlo simulations.

The measurement models for the scalar and vector tracking architectures were substituted into general expressions of LS, WLS, EKF, UKF, PF methods. Both CRLB and posterior CRLB were derived, as well. Next, we derived the *scalar iterative EKF for the PVT problem* and a similiar method for approximation of the WLS method based on the FG framework.

By extensive simulations, we proved that the FG-based *scalar iterative EKF can be used for the PVT estimation provided the number of visible SVs reaches at least sixteen. The accuracy is comparable to that of the EKF at three iterations or more and depends on the dynamics of the user, the number of iterations and filter's driving*

noise variance. It was observed that the filter performs better if the higher dynamics are not modeled with high driving noise variance, but the number of iterations is increased instead. The filter thus offers compromises in complexity and accuracy. If the number of SVs is high or the number of iterations is increased in dynamic scenarios, the scalar EKF can even outperform the standard vector EKF. In dynamic scenarios, the iterations help in the way that a new linearizing point is established for each new iteration.

The numbers of flops of both scalar EKF and the standard EKF were comparable if an efficient form of the EKF was implemented. For a single iteration, which is sufficient for low dynamics, the scalar EKF has lower number of flops, for two iterations it is comparable and for three iterations, the EKF has lower number of flops. We should not deduce that the EKF is less complex even if its number of flops is lower, since the vector and matrix manipulations involve memory reordering, unlike the distributed scalar EKF operating on scalars.

To account for the correlation of the pseudorange and pseudorange rate measurements between the satellite channels and other atmospheric biases, another case-study-based simulations were conducted. At the same time, we showed that the FG-based filter can be adopted to the vector tracking architecture with comparable performance to the EKF. *The 7 dB tracking sensitivity benefit of the vector tracking architecture is lowered to 5dB.*

By experiment with WNav receiver, we demonstrated that a comparable precision to the EKF can be achieved.

To sum up the study, we recall that *the algorithm enables engineers* to offload the CPU into the hardware logic (HW) and hence:

- employ a low-end CPU and tradeoff the HW logic with the CPU
- track a higher number of SVs in the HW logic
- by faster updates of the PVT estimates and predictions, improve the stability of the vector tracking architecture and in some applications eliminate the need of inertial sensors.

If the algorithm is implemented in the CPU, the following advantages have been identified:

- inclusion of the matrix library can be avoided resulting in lower code size
- by storing only the actual mean and variance, the requirements on the data memory reduce.

Bibliography

[1] Global Positioning System Wing (GPSW) Systems Engineering & Integration, *Interface Specification IS-GPS-200 - Navstar GPS Space Segment/Navigation User Interface*, June 2010, Revision E.

[2] ——, *Interface Specification IS-GPS-705 - Navstar GPS Space Segment/User Segment L5 Interface*, June 2010, Revision A.

[3] ——, *Interface Specification IS-GPS-800 - Navstar GPS Space Segment/User Segment L1C Interface*, June 2010, Revision A.

[4] Russian Institute of Space Device Engineering, *Global Navigation Satellite System GLONASS - Interface Control Document - Navigation Radiosignal in Bands L1, L2*, 5th ed., 2008.

[5] European Space Agency (ESA), *European GNSS (Galileo) Open Service - Signal In Space Interface Control Document*, September 2010, issue 1.1.

[6] China Satellite Navigation Office, *BeiDou Navigation Satellite System - Signal in Space - Interface Control Document*, 2011, test version.

[7] Department of Transportation United States of America and Federal Aviation Administration, *Global Positioning System Wide Area Augmentation System WAAS Performance Standard*, 1st ed., October 2008.

[8] RTCA SC-159, *Minimum Operational Performance Standards for Global Positioning System/Wide Area Augmentation System Airborne Equipment*, December 2006, DO-229D.

[9] European Space Agency (ESA), *User Guide for European Geostationary Navigation Overlay Service EGNOS Application Developers*, 1st ed., July 2009.

[10] European Commision - Directorate-General for Energy and Transport, *EGNOS Service Definition Document - Open Service*, 1st ed., October 2009.

[11] Japan Aerospace Exploration Agency, *Quasi-Zenith Satellite System - Navigation Service*, 1st ed., February 2012.

[12] P. Misra, P. Enge, *Global Positioning System: Signals, Measurements, and Performance.*, 1st ed. Massachusetts: Ganga-Jamuna Press, 2001.

[13] E. D. Kaplan and C. J. Hegarty, *Understanding GPS: Principles and Applications*, 2nd ed. Norwood, MA: Artech House, Inc., 2006.

[14] T. Pany, *Navigation Signal Processing for GNSS Software Receiver - GNSS Technology and Applications*, 1st ed. Norwood, MA, USA: Artech House, January 2010.

[15] C. Stober et al., "ipexSR: A real-time multi-frequency software GNSS receiver," in *Proceedings of ELMAR*, Zadar, Croatia, September 2010, pp. 407 – 416.

[16] O. Jakubov, P. Kovar, P. Kacmarik, and F. Vejrazka, "The Witch Navigator - a low cost GNSS software receiver for advanced processing techniques," *Radioengineering*, vol. 19, no. 4, pp. 536 – 543, December 2010.

[17] N. Abassian and M. G. Petovello, "Implementation of dual-frequency GLONASS and GPS L1 C/A software receiver," *The Journal of Navigation*, no. 63, pp. 269 – 287, 2010.

[18] H. Afzal and G. Lachapelle, "Design methodology for a dual frequency configurable GPS receiver," in *Proceedings of GNSS10*, Portland, OR, September 2010, pp. 1 – 9.

[19] N. C. Shivaramaiah and A. G. Dempster, *Global Navigation Satellite Systems - Signal, Theory and Applications: Chapter 2 Baseband Hardware Design in Modernised GNSS Receivers*, 1st ed. IntechOpen, 2012.

[20] J. Mandel, "Efficient implementation of the Ensemble Kalman Filter," Center for Computational Mathematics Reports, University of Colorado at Denver and Health Sciences Center, Denver, CO 80217-3364, May 2006.

[21] H. C. Godinez and J. D. Moulton, "An efficient matrix-free ensemble Kalman filter," Los Alamos National Laboratory of ADTSC Technical Report, February 2011.

[22] E. Wan and R. V. der Merwe, "The unscented Kalman filter for nonlinear estimation," in *IEEE Symposium on Adaptive Systems for Signal Processing, Communications and Control*, April 2000, pp. 604 – 611.

[23] T. Bo, C. Pingyuan, and C. Yangzhou, "Sigma-point Kalman filters for GPS based position estimation," in *Information, Communications and Signal Processing, 2005 Fifth International Conference on*, 2005, pp. 213 – 217.

[24] A. Doucet, N. Freitas, and N. Gordon, *An Introduction to Sequential Monte Carlo Methods in Sequential Monte Carlo Methods in Practice*, 1st ed. New York: Springer, 2001.

[25] P. M. Djuric, *Sequential Monte Carlo Methods in Practise: Chapter Sequential Estimation of Signals under Model Uncertainty*, 1st ed. Springer-Verlag, 2001.

[26] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transaction on Signal Processing*, vol. 50, no. 2, pp. 174 – 188, 2002.

[27] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425 – 437, Feb. 2002.

[28] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transaction on Information Theory*, vol. 47, no. 2, pp. 498 – 519, February 2004.

[29] H. A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Magazine*, pp. 28 – 41, January 2004.

[30] J. C. Chen, C. S. Maa, and Y. C. Wang, "Mobile position location using factor graphs," *IEEE Communications Letters*, vol. 7, no. 9, pp. 431 – 433, September 2003.

[31] J. C. Chen, Y. C. Wang, C. S. Maa, and J. T. Chen, "Network-side mobile position location using factor graphs," *IEEE Transaction on Wireless Communications*, vol. 5, no. 10, pp. 2696 – 2704, October 2006.

[32] C. Mensing and S. Plass, "Positioning based on factor graphs," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, pp. 1 – 11, 2007, article ID 41348.

[33] J. C. Chen, P. Ting, C. S. Maa, and J. T. Chen, "Wireless geolocation with TOA/AOA measurements using factor graph and the sum-product algorithm," in *Proceedings of the 60th IEEE Vehicular Technology Conference (VTC '04)*, Los Angeles, CA, September 2004, pp. 3526 – 3529.

[34] H. Liu, F. K. W. Chan, and H. C. So, "Non-line-of-sight mobile positioning using factor graphs," *IEEE Transaction on Vehicular Technology*, vol. 58, no. 9, pp. 5279 – 5283, November 2009.

[35] H. Wymeersch, J. Lien, and Z. W. Moe, "Cooperative localization in wireless networks," *Proceedings of the IEEE*, vol. 97, no. 2, pp. 2696 – 2704, February 2009.

[36] M. A. Caceres, F. Penna, H. Wymeersch, and R. Garello, "Hybrid GNSS-terrestrial cooperative positioning via distributed belief propagation," in *Proceedings of GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*. Chalmers Publication Library, 2010, pp. 1 – 5.

[37] O. Jakubov, P. Kacmarik, P. Kovar, and F. Vejrazka, "Reduced-complexity GNSS software simulator based on correlator output signal modeling," in *Proceedings of 18th Saint Petersburg International Conference on Integrated Navigation Systems*, St. Petersburg, Russia, May 2011, pp. 348 – 354.

[38] P. Kacmarik, P. Kovar, O. Jakubov, and F. Vejrazka, "The Witch Navigator - a software GNSS receiver built on real-time Linux," in *Proceedings of 13th Real-Time Linux Workshop*, Prague, Czech rep., October 2011, pp. 17 – 28.

[39] P. Kovar, P. Kacmarik, O. Jakubov, and F. Vejrazka, "The Witch Navigator - a low cost software receiver for education and research," in *Proceedings of European Navigation Conference ENC2011*, London, United Kingdom, November 2011, pp. 1 – 10.

[40] ——, "The Witch Navigator - a software GNSS receiver for education and research," in *Proceedings of IGNSS 2011 Conference & Exhibition*, Sydney, Australia, November 2011, pp. 1 – 11.

[41] P. Kovar, P. Kacmarik, and F. Vejrazka, "Economic Galileo E5 receiver," *Radioengineering*, vol. 21, no. 1, pp. 346 – 355, April 2012.

[42] P. Kovar, P. Kacmarik and F. Vejrazka, "Low complex interoperable GNSS signal processor and its performance," in *Proceedings of IEEE/ION PLANS*

2010 Position Location and Navigation Symposium, Palm Springs, CA USA, May 2010, pp. 947 – 951.

[43] S. M. Kay, *Fundamentals of Statistical Signal Processing: Volume I: Estimation Theory*, 1st ed. New Jersey: Prentice-Hall, 1993.

[44] O. Jakubov, Kacmarik, P. Kovar, P., and F. Vejrazka, "Reduced-complexity GNSS software simulator based on correlator output signal modeling," in *18th Saint Petersburg International Conference on Integrated Navigation Systems*, St. Petersburg, Russia, May/June 2011, pp. 378 – 380.

[45] O. Jakubov, P. Kacmarik, P. Kovar, and F. Vejrazka, "Universality and realistic extensions to the semi-analytic simulation principle in GNSS signal processing," *Radioengineering*, vol. 21, no. 2, pp. 647 – 658, June 2012.

[46] P. Kovar, P. Kacmarik, O. Jakubov, and F. Vejrazka, "The implementation of dual frequency GLONASS to the Witch Navigator," in *Proceedings of 18th Saint Petersburg International Conference on Integrated Navigation Systems*, St. Petersburg, Russia, May 2011, pp. 378 – 380.

[47] O. Jakubov, P., and P. Kovar, "Signal processing algorithms in gps, galileo, and glonass integrated receiver," in *Proceedings of GNSS PhD Summit 2011*, Munich, Germany, November 2011, pp. 1 – 9.

[48] O. Jakubov, P. Kovar, P. Kacmarik, and F. Vejrazka, "Distributed extended kalman filter for position, velocity, time estimation in satellite navigation receivers," *Radioengineering*, pp. 1 – 15, May 2013, submitted for review.

[49] H. A. Loeliger, "Least squares and Kalman filtering on Forney graphs," in *Codes, Graphs, and Systems*. Kluwer, 2002, pp. 113 – 135.

[50] J. Proakis, *Digital Communications*, 4th ed. McGraw-Hill Science/Engineering/Math, August 2000.

[51] M. K. Simon, J. K. Omura, R. A. Scholtz, and B. K. Levitt, *Spread Spectrum Communications Handbook*. McGraw-Hill Science/Engineering/Math, April 1994, Rev. Sub.

[52] S. M. Kay, *Fundamentals of Statistical Signal Processing: Volume II: Detection Theory*, 1st ed. New Jersey: Prentice-Hall, 1998.

[53] A. G. Meyr, H., *Digital Communication Receivers: Synchronization in Digital Communications. Volume I. Phase-, Frequency-Locked Loops and Amplitude Control.* Norwood: John Wiley & Sons, Inc., 1990.

[54] M. F. S. A. Meyr, H. Moeneclaye, *Digital Communication Receivers. Synchronization, Channel Estimation and Signal Processing.* New York: John Wiley & Sons, Inc., 1998.

[55] J. H. Won, D. Dötterböck, and B. Eissfeller, "Performance comparison of different forms of Kalman filter approaches for a vector-based GNSS signal tracking loop," *Navigation*, vol. 57, no. 3, pp. 185–199, Fall 2010.

[56] D. U. Sanli and S. Tekic, *Accuracy of GPS Precise Point Positioning: A Tool for GPS Accuracy Prediction*, 1st ed. LAP Lambert Academic Publishing, Aprilr 2010.

[57] J. C. Cohenour, *Global Positioning System Clock and Orbit Statistics and Precise Point Positioning*, 1st ed. ProQuest, UMI Dissertation Publishing, September 2011.

[58] J. W. Betz, "Binary offset carrier modulations for radionavigation," *Journal of the Institute of Navigation*, vol. 48, no. 4, pp. 227–245, Winter 2001-2002.

[59] N. C. Shivaramaiah and A. G. Dempster, "The Galileo E5 AltBOC: Understanding the signal structure," in *IGNSS Symposium 2009.* Qld, Australia: International Global Navigation Satellite Systems Society, December 2009, pp. 1 – 13.

[60] E. S. Lohan, A. Lakhzouri, and M. Renfors, "Complex double-binary-offset-carrier modulation for a unitary characterisation of Galileo and GPS signals," in *IEE Proceedings - Radar Sonar Navigation*, vol. 153, no. 3, October 2006, pp. 403 – 408.

[61] J.-M. Sleewaegen, W. D. Wilde, and M. Hollreiser, "Galileo AltBOC receiver," [Online] http://www.septentrio.com/content/galileo-altboc-receiver [quoted 2010-05-01], pp. 1 – 9, May 2004.

[62] P. Kacmarik, P. Kovar, and F. Vejrazka, "Galileo AltBOC E5 signal characteristics for optimal tracking algorithms," in *Proceedings of NAV08/ILA37 The Navigation Conference & Exhibition*, London: Royal Institute of Navigation, 2008, pp. 1 – 9.

[63] P. Kovar, P. Kacmarik, and F. Vejrazka, "High performance Galileo E5 correlator design," in *Proceedings of 13th IAIN World Congress*, Stockholm: Nordic Institute of Navigation, 2009, pp. 1 – 8.

[64] A. J. V. Dierendonck, P. Fenton, and T. Ford, "Theory and performance of narrow correlator spacing in GPS receiver," *Navigation: J. Inst. Navigation*, vol. 39, no. 3, pp. 265 – 283, Fall 1992.

[65] R. D. J. van Nee, J. Siereveld, P. C. Fenton, and B. R. Townsend, "The multipath estimating delay lock loop: Approaching theoretical accuracy limits," in *The IEEE Position Location and Navigation Symposium*, Las Vegas, Nevada, USA, April 1994, pp. 921 – 921.

[66] P. Closas, "Bayesian signal processing techniques for GNSS receivers: from multipath mitigation to positioning," Ph.D. dissertation, Universitat Politecnica de Catalunya, Barcelona, 2009.

[67] P. Closas, C. Fernandez-Prades, and J. A. Fernandez-Rubio, "A Bayesian approach to multipath mitigation in GNSS receivers," *IEEE Journal of Selected Topics in Signal Processing*, vol. 3, no. 4, pp. 1 – 12, August 2009.

[68] F. Amoroso, "Adaptive A/D converter to suppress CW interference in DSPN spread-spectrum communications," *IEEE Transactions on Communications*, vol. 31, no. 10, pp. 1117 – 1123, October 1983.

[69] K. M. Chugg and M. Zhu, "A new approach to rapid PN code acquisition using iterative message passing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 5, pp. 884 – 897, May 2005.

[70] O. W. Yeung and K. M. Chugg, "An iterative algorithm and low complexity hardware architecture for acquisition of long PN codes in UWB systems," *Journal of VLSI Signal Processing*, no. 43, pp. 25 – 42, 2006.

[71] S. H. Won and L. Hanzo, "Iterative code acquisition for DS-UWB downlink using multiple-component decoders," *Electronic Letters*, vol. 44, no. 2, pp. 1 – 2, 2008.

[72] F. Principe, K. M. Chugg, and M. Luise, "Rapid acquisition of Gold codes and related sequences using iterative message passing on redundant graphic models," in *MILCOM'06 Proceedings of the 2006 IEEE conference on Military communications*, Washington, D.C., 2008, pp. 1483 – 1489.

[73] E. M. Copps, G. J. Geier, W. C. Fidler, and P. A. Grundy, "Optimal processing of GPS signals," *Navigation*, vol. 27, no. 3, pp. 171 – 182, 1980.

[74] J. W. Sennott and D. Senffner, "The use of satellite geometry for prevention of cycle slips in a GPS processor," *Navigation*, vol. 39, no. 2, pp. 217–236, 1992.

[75] ——, "Comparison of continuity and integrity characteristics for integrated and decoupled demodulation/navigation receivers," in *Proceedings of the 8th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS 1995)*, Palm Springs, CA, September 1995, pp. 1531 – 1537.

[76] J. Spilker, "Vector delay lock loop processing of radiolocation transmitter signals," U.S. Patent 5 398 034, March 14, 1995. [Online]. Available: http://www.freepatentsonline.com/5398034.html

[77] A. Jovancevic, G. S. Brown, A., J. Noronha, and B. Sirpatil, "Ultra-tight coupling implementation using real time software receiver," in *Proceedings of the 17th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2004)*, Long Beach, CA, 2004, pp. 1575 – 1586.

[78] E. J. Ohlmeyer, "Analysis of an ultra-tightly coupled GPS/INS system in jamming," in *Proceedings of IEEE/ION PLANS 2006*, San Diego, CA, April 2006, pp. 44 – 53.

[79] M. G. Petovello and G. Lachapelle, "Comparison of vector-based software receiver implementations with application to ultra-tight GPS/INS integration," in *Proceedings of the 19th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2006)*, Forth Worth, TX, September 2006, pp. 1790 – 1799.

[80] R. E. B. Pany, T. Kaniuth, "Deep integration of navigation solution and signal processing," in *Proceedings of the 18th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION-GNSS 2005)*, Long Beach, CA, September 2005, pp. 1095 – 1102.

[81] B. Pany, T. Eissfeller, "Use of vector delay lock loop receiver for GNSS signal power analysis in bad signal conditions," in *Proceedings of IEEE/ION PLANS 2006*, San Diego, CA, April 2006, pp. 893 – 903.

[82] M. Lashley and D. M. Bevly, "Vector delay/frequency lock loop implementation and analysis," in *Proceedings of the 2009 International Technical Meeting of of the Institute of Navigation*, Anaheim, CA, January 2009, pp. 1073 – 1086.

[83] M. Lashley, D. M. Bevly, and J. Y. Hung, "Performance Analysis of Vector Tracking Algorithms for Weak GPS Signals in High dynamics," *IEEE Journal of Selected Topics in Signal Processing*, vol. 3, no. 4, pp. 661 – 673, August 2009.

[84] D. Benson, "Interference benefits of a vector delay lock loop (VDLL) GPS receiver," in *Proceedings of 63rd Annual Meeting of the Institute of Navigation*, Cambridge, Massachusetts, April 2007, pp. 1 – 8.

[85] S. Bhattacharyya and D. Gebre-Egziabher, "Development and validation of parametric models for vector tracking loops," *Navigation*, vol. 57, no. 4, pp. 275 – 295, Winter 2010.

[86] P. Closas, C. Fernandez-Prades, and J. A. Fernandez-Rubio, "Maximum likelihood estimation of position in GNSS," *IEEE Signal Processing Letters*, vol. 14, no. 5, pp. 359 – 361, May 2007.

[87] ——, "Direct position estimation approach outperforms conventional two-step positioning," in *Proceedings of 17th European Signal Processing Conference*, Glasgow, Scotland, August 2009, pp. 1 – 5.

[88] P. Closas, C. Fernandez-Prades, and J. Fernandez-Rubio, "Cramer-Rao bound analysis of positioning approaches in GNSS receivers," *Signal Processing, IEEE Transactions on*, vol. 57, no. 10, pp. 3775 – 3786, oct. 2009.

[89] P. Closas, C. Fernandez-Prades, D. Bernal, and J. A. Fernandez-Rubio, "Bayesian direct position estimation," in *Proceedings of ION GNSS 21st International Technical Meeting of the Satellite Division*, Savannah, GA, September 2008, pp. 183 – 190.

[90] P. Closas, C. Fernandez-Prades, and J. A. Fernandez-Rubio, "Direct position estimation approach outperforms conventional two-steps positioning," in *Proceedings of 17th European Signal Processing Conference (EUSIPCO 2009)*, Glasgow, Scotland, 2009, pp. 1958 – 1962.

[91] Xilinx All Programmable, "Zynq-7000 extensible processing platform," [Online] http://www.xilinx.com/products/silicon-devices/epp/zynq-7000/index.htm [quoted 2012-05-03], May 2012.

[92] M. G. Petovello, C. O'Driscoll, G. Lachapelle, D. Borio, and H. Murtaza, "Architecture and benefits of an advanced GNSS software receiver," *Journal of Global Positioning Systems*, vol. 7, no. 2, pp. 156 – 168, 2008.

[93] T. Hobiger, T. Gotoh, J. Amagai, Y. Koyama, and T. Kondo, "A GPU based real-time GPS software receiver," *GPS Solutions*, vol. 14, pp. 207 – 216, 2010.

[94] C. Wu, Y. Qian, X. Cui, and M. Lu, "The optimized method and algorithms in the CPU&GPU-based GNSS software receiver," in *Proceeding of the 22nd International Technical Meeting of the Satellite Divition of the Institute of Navigation (ION GNSS 2009)*, Savannah, GA, September 2009, pp. 339 – 343.

[95] A. Knežević, C. O'Driscoll, and G. Lachapelle, "Co-processor aiding for real-time software GNSS receivers," in *ION ITM 2010*, San Diego, CA, USA, January 2010, pp. 1 – 12.

[96] J. Seo, Y.-H. Chen, D. S. D. Lorenzo, S. Lo, P. Enge, D. Akos, and J. Lee, "A real-time capable software-defined receiver using GPU for adaptive anti-jam GPS sensors," *Sensors*, vol. 11, pp. 8966 – 8991, September 2011.

[97] I. Buck, "The evolution of GPUs for general purpose computing," in *GPU Technology Conference 2010*, San Jose, CA, USA, September 2010, pp. 20 – 23.

[98] P. J. Mumford, K. Parkinson, and A. Dempster, "The Namuru open GNSS research receiver," in *Proceeding of the 19th International Technical Meeting of the Satellite Divition of the Institute of Navigation (ION GNSS 2006)*, Fort Worth, TX, September 2006, pp. 2847 – 2855.

[99] P. J. Mumford and Y. Heo, "Namuru-GPL, open source software for the Namuru FPGA-based GNSS receiver," 2007.

[100] P. Kovar, P. Kacmarik and F. Vejrazka, "Universal front end for software GNSS receiver," in *Proceedings of 13th IAIN World Congress*, Bergen, Norway, October 2009, pp. 1 – 6.

[101] S. U. Papoulis, A. Pillai, *Probability, Random Variables and Stochastic Processes*, 4th ed. New York: McGraw Hill, 2002.

[102] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramer-Rao bounds for discrete-time nonlinear filtering," *IEEE Transaction on Signal Processing*, vol. 46, no. 5, pp. 1386 – 1396, May 1998.

[103] O. Shental, P. H. Siegel, J. K. Wolf, D. Bickson, and D. Dolev, "Gaussin belief propagation solver for systems of linear equations," in *ISIT 2008, IEEE International Symposium on Information Theory 2008*, Toronto, ON, July 2008, pp. 1863 – 1867.

[104] D. Bickson, O. Shental, and D. Dolev, "Distributed Kalman filter via Gaussian belief propagation," in *The 46th Annual Allerton Conference on Communication, Control and Computing*, Allerton, Illinois, September 2008, pp. 1 – 3.

[105] D. S. Watkins, *Fundamentals of Matrix Computations*, 2nd ed. New York: John Wiley & Sons, March 2002.

[106] D. Borio, P. B. Anantharamu, and G. Lachapelle, "SATLSim: a semi-analytic framework for fast GNSS tracking loop simulations," *GPS Solutions*, pp. 1 – 5, May 2011.

Appendix A

Complexity of Matrix and Vector Operations

The complexity in terms of flops for basic matrix and vector operations is summarized in Table A.1. The matrix inversion lemma supposed in the table is as follows.

Assume $\mathbf{A}$ as an $n \times n$ matrix, $\mathbf{B}$ is $n \times m$, $\mathbf{C}$ is $m \times m$, $\mathbf{D}$ is $m \times n$, and that the indicated inverses exist. The following identity then holds

$$(\mathbf{A} + \mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{DA}^{-1}\mathbf{B} + \mathbf{C}^{-1}\right)^{-1}\mathbf{DA}^{-1}. \tag{A.1}$$

Table A.1: Number of flops for various matrix and vector operations, taken from [105]

Operation(s)	Flops	Notes
$\mathbf{x}^T\mathbf{y}$	$2n-1$	$\mathbf{x}$, $\mathbf{y}$ are $n \times 1$
$\mathbf{x}+\mathbf{y}$, $\alpha\mathbf{x}$	$2n$	$\alpha \in \mathbb{R}$
$\mathbf{Ax}$	$m(2n-1)$	$\mathbf{A}$ is $m \times n$
$\mathbf{AB}$	$mp(2n-1)$	$\mathbf{B}$ is $n \times p$
$\mathbf{D}^{-1}\mathbf{x}$	n	$\mathbf{D}$ is $n \times n$ and diagonal
$\mathbf{T}^{-1}\mathbf{x}$	n^2	$\mathbf{T}$ is $n \times n$ and triangular
$\mathbf{G}^{-1}$	$2n^2$	$\mathbf{G}$ is $n \times n$ and orthogonal
$\mathbf{A}=\mathbf{PLU}$	$\frac{2}{3}n^3$	$\mathbf{A}$ is nonsingular
		$\mathbf{P}$ is a permutation matrix
		$\mathbf{L}(\mathbf{U})$ is a lower (upper) triangular matrix
$\mathbf{A}=\mathbf{LL}^T$	$\frac{1}{3}n^3$	$\mathbf{A}$ is positive definite
	$\frac{1}{3}n^3+2n^2$	when solving linear equations with LU or Choleski decomposition
$(\mathbf{A}+\mathbf{BD})^{-1}$	$\frac{2}{3}n^3+2pn^2$	$\mathbf{A}$ is $n \times n$, $\mathbf{B}$ is $n \times p$, $\mathbf{D}$ is $p \times n$
$(\mathbf{A}+\mathbf{BD})^{-1}$	$2p^2n+\frac{2}{3}p^3$	using matrix inversion lemma (A.1) for $\mathbf{C}=\mathbf{I}$
		$(\mathbf{A}+\mathbf{BD})^{-1}=\mathbf{A}^{-1}-\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{DA}^{-1}\mathbf{B}+\mathbf{I}\right)^{-1}\mathbf{DA}^{-1}$