



Degree Project in Mathematical Statistics

Second Cycle, 30 credits

Bayesian Estimation of Sea Clutter Parameters for Radar

A Stochastic Approach

EMMA ÖIJAR JANSSON

Abstract

Radars operating at sea encounter a common phenomenon known as sea clutter, characterized by undesired reflections originating from the sea surface. This phenomenon can significantly impair the radar's capacity to detect small, slow-moving targets. Therefore, it is crucial to gain a comprehensive understanding of the statistical attributes that describes the sea clutter. This comprehension is pivotal for the development of efficient signal processing strategies.

The core of this work revolves around the imperative requirement for accurate statistical models to characterize sea clutter. Within this context, this work particularly explores the application of Field's model. Field's model describes the sea clutter process using three stochastic differential equations that form the dynamical process of the complex reflectivity of the sea surface. One equation describes the radar cross section, which is given by a Cox-Ingersoll-Ross process, parameterized by the parameters \mathcal{A} and α . The other equations describe the speckle process, which is a complex Ornstein-Uhlenbeck process parameterized by \mathcal{B} . The aim of this thesis is to explore the possibilities in estimating the parameters \mathcal{A} , α and \mathcal{B} in Field's model through the application of Bayesian inference.

To achieve this objective, Metropolis-Hastings and Sequential Monte Carlo methods are employed. The clutter data, represented by the complex reflectivity, is synthetically generated by using the Euler-Maruyama and Milstein schemes. Three algorithms are designed for estimating the sea clutter parameters. Two algorithms require 300 seconds of data and are based on the approach suggested by Clement Roussel in his PhD thesis [1]. Specifically, these algorithms employ the Metropolis-Hastings method for estimating \mathcal{A} , α and \mathcal{B} , respectively. As input data to the algorithms, estimators of the Cox-Ingersoll-Ross process and the real part of the Ornstein-Uhlenbeck process are utilized. In contrast, the last algorithm describes an approach that employs only 3 seconds of data. This algorithm is a Metropolis-Hastings method that incorporates a particle filter for approximation of likelihoods.

For evaluation of the algorithms, two distinct sets of parameters are considered, leading to varying characteristics of the complex reflectivity. The two algorithms that require 300 seconds of data are executed ten times for each parameter set. Evidently, the algorithm designed for estimating \mathcal{B} generates values that closely aligns with the true values while the algorithm designed for estimating \mathcal{A} and α does not yield as satisfactory results. Due to time constraints and the computational demands of the simulations, the last algorithm, requiring 3 seconds of data, is executed only twice for each parameter set. Remarkably, this algorithm generates estimates that agree with the true values, indicating strong performance. Nonetheless, additional simulations are required to conclusively confirm its robustness.

To conclude, it is possible to estimate sea clutter parameters within Field's model by using the applied methods of Bayesian inference. However, it is important to analyze the applicability of these methods for a large quantity of diverse clutter data. Moreover, their computational demands pose challenges in real-world applications. Future research should address the need for more computationally efficient methods to overcome this challenge.

Byesiansk estimering av sjöklutterparametrar för radar

en stokastisk approach

Sammanfattning

Radar som verkar till havs behöver hantera ett fenomen som kallas för sjöklutter, vilket är oönskade reflektioner från havsytan. Detta fenomen kan avsevärt försämra radarns förmåga att upptäcka långsamt rörliga mål. Det är därför viktigt att erhålla en förståelse för den statistik som beskriver sjökluttret. Denna förståelse är avgörande för utvecklingen av effektiva signalbehandlingsstrategier.

Detta arbete fokuserar på den viktiga aspekten av att använda korrekta statistiska modeller för att beskriva sjöklutter. Specifikt undersöker detta arbete Field's modell, som beskriver den komplexa reflektiviteten från havsytan med hjälp av tre stokastiska differentialekvationer. En ekvation beskriver radarmålarean (radar cross section) som är en Cox-Ingersoll-Ross-process, parametriserad av \mathcal{A} och α . De andra ekvationerna beskriver speckle-processen som är en komplex Ornstein-Uhlenbeck-process, parametriserad av \mathcal{B} . Syftet med denna uppsats är att utforska möjligheter för att estimeras parametrarna \mathcal{A} , α och \mathcal{B} i Field's modell genom tillämpning av Bayesiansk inferens.

För att uppnå detta, används Metropolis-Hastings-algoritmer samt sekventiella Monte-Carlo-metoder. Klotterdatan som representeras av den komplexa reflektiviteten genereras med hjälp av Euler-Maruyama- och Milstein-scheman. Sammanlagt designas tre algoritmer för att estimeras sjöklutterparametrarna. Två algoritmer behöver 300 sekunder av data och är baserade på tidigare arbeten av C. Rousell [1]. Dessa algoritmer använder Metropolis-Hastings för att uppskatta \mathcal{B} , respektive \mathcal{A} och α . Som indata till algoritmerna används estimatorer för Cox-Ingersoll-Ross-processen samt den reella delen av Ornstein-Uhlenbeck-processen. Den sista algoritmen beskriver istället ett tillvägagångssätt som endast kräver 3 sekunders data. Denna algoritm är en Metropolis-Hastings-algoritm som använder ett partikelfilter för approximering av likelihoods.

För utvärdering av algoritmerna beaktas två olika parameteruppsättningar, vilka genererar olika komplexa reflektiviteter. De två algoritmerna som kräver 300 sekunder av data körs tio gånger för varje parameteruppsättning. Algoritmen designad för att uppskatta \mathcal{B} genererar värden som är nära de sanna värdena medan algoritmen designad för att uppskatta \mathcal{A} och α inte ger lika tillfredsställande resultat. På grund av tidsbrist och den långa simuleringstiden, körs den sista algoritmen, som kräver 3 sekunder av data, endast två gånger för varje parameteruppsättning. Anmärkningsvärt är att denna algoritm genererar uppskattningar som faktiskt stämmer ganska väl med de sanna värdena, vilket indikerar på stark prestanda. Dock krävs ytterligare simuleringar för att bekräfta detta.

Sammanfattningsvis är det möjligt att uppskatta sjöklutterparametrarna i Field's modell med de Bayesianska inferensmetoderna som tillämpas i detta arbete. Det är dock viktigt att beakta hur användbara dessa metoder är för en variation av klotterdata. Dessutom innebär den långa beräkningstiden utmaningar i verkliga tillämpningar. Framtida studier bör adressera behovet av mer beräkningsmässigt effektiva metoder för att övervinna denna utmaning.

Acknowledgement

First and foremost, I am extremely grateful to Saab for the opportunity to undertake this thesis and the invaluable experience I gained during my time there. I would like to express my deepest appreciation to my supervisors Alexander Lindmaa and Adam Andersson from Saab for their valuable support and feedback. They have generously provided knowledge and expertise throughout this entire process. Additionally, I want to thank my examiner and supervisor from KTH, Pierre Nyquist.

I am also deeply grateful to my boyfriend Jörgen Smit, for his feedback and emotional support during this challenging time. I extend my heartfelt thanks for his generosity in lending me his computer when mine was unable to meet the demands of this project.

Lastly, I would like to acknowledge my family and friends, especially my parents and twin. Their emotional support and motivation have truly encouraged me to complete this work.

Stockholm, September 2023

Emma Öijar Jansson

Author

Emma Öijar Jansson <eoj@kth.se>
Degree Program in Applied and Computational Mathematics
KTH Royal Institute of Technology

Place for Project

Linköping, Sweden
Saab Dynamics

Supervisors at Saab

Alexander Lindmaa <alexander.lindmaa@saabgroup.com>, Saab Dynamics
Adam Andersson <adam.andersson2@saabgroup.com>, Saab Surveillance

Supervisor and Examiner at KTH

Pierre Nyquist <pierren@kth.se>

List of Algorithms

1	The Metropolis-Hastings Algorithm	14
2	The Adaptive Metropolis-Hastings Algorithm	15
3	Sequential Importance Sampling (SIS)	18
4	Metropolis-Hastings for Estimating \mathcal{B}	26
5	Metropolis-Hastings for Estimating \mathcal{A} and α	28
6	Particle Filter for Likelihood Approximation	30
7	Particle Filter within Metropolis-Hastings for Estimating \mathcal{A} , α and \mathcal{B}	32
8	Sequential Importance Resampling (SIR)	54
9	The Bootstrap Filter	55

List of Tables

6.1	<i>Constants for Synthetic Data.</i>	24
6.2	<i>Duration time T and corresponding average \bar{B}_Ψ for 20 trajectories.</i>	25
7.1	<i>Sets of parameter values from which the observed Reflectivity is produced.</i>	37
7.2	<i>Variables used in the simulation process for estimating \mathcal{B} with Algorithm 4.</i>	38
7.3	<i>Estimated mean of the Markov chain for \mathcal{B} generated by Algorithm 4 after the burn-in for ten simulations and two different parameter sets.</i>	39
7.4	<i>Variables used in the simulation process for estimating \mathcal{A} and α with Algorithm 5.</i>	39
7.5	<i>Estimated mean of the Markov chains for \mathcal{A} and α generated by Algorithm 5 after the burn-in for ten simulations and two different parameter sets.</i>	41
7.6	<i>Values for N, T and σ.</i>	42
7.7	<i>Initial values and variables serving as inputs to Algorithm 6 and 7.</i>	45
7.8	<i>Estimated mean of the Markov chains for \mathcal{A}, α and \mathcal{B} generated by Algorithm 7 after the burn-in for two simulations and set 1 of true parameters.</i>	48
7.9	<i>Estimated mean of the Markov chains for \mathcal{A}, α and \mathcal{B} generated by Algorithm 7 after the burn-in for two simulations and set 2 of true parameters.</i>	48

List of Figures

3.1	<i>Sample path of an Ornstein-Uhlenbeck process, with $\mu = 0.5$, $\theta = 0.1$, $\sigma = 0.1$ and initial value $S_0 = 2$. The black line corresponds to the long term mean μ.</i>	9
3.2	<i>Sample path of a Cox-Ingersoll-Ross process, with $\theta = 0.5$, $k = 0.1$, $\sigma = 0.1$ and initial value $S_0 = 2$. The black line corresponds to the long term mean θ.</i>	10
4.1	<i>Hidden Markov Model for a recursive state estimation problem.</i>	16
7.1	<i>Comparison between x_t and the estimators $\bar{x}_t, \hat{x}_t, \tilde{x}_t$ for a segment comprising 4.5 seconds out of a total of 300 seconds. (a) Comparison for a single sample path. (b) Comparison of the average obtained from 20 trajectories.</i>	35
7.2	<i>The RMSE for the three estimators over 4.5 seconds.</i>	35
7.3	<i>Comparison between $\gamma_t^{(R)}$ and the estimator $\tilde{\gamma}_t^{(R)}$ for a segment comprising 0.2 seconds out of a total of 300 seconds. (a) Comparison for a single sample path. (b) Comparison of the average obtained from 20 trajectories.</i>	36
7.4	<i>The RMSE for $\tilde{\gamma}_t^{(R)}$ over 0.2 seconds.</i>	37
7.5	<i>Markov chain and distribution of \mathcal{B} for one simulation with Algorithm 4. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{B}. (b) Distribution of \mathcal{B} after burn-in.</i>	38
7.6	<i>Markov chain and distribution of \mathcal{B} for one simulation with Algorithm 4. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. (a) Markov chain for \mathcal{B}. (b) Distribution of \mathcal{B} after burn-in.</i>	39
7.7	<i>Markov chains and distributions of \mathcal{A} and α for one simulation with Algorithm 5. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{A}. (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α. (d) Distribution of α after burn-in.</i>	40
7.8	<i>Markov chains and distributions of \mathcal{A} and α for one simulation with Algorithm 5. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. (a) Markov chain for \mathcal{A}. (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α. (d) Distribution of α after burn-in.</i>	41
7.9	<i>Kernel density estimations compared with particle distributions.</i>	43
7.10	<i>The real part of the complex reflectivity $\Psi^{(R)}$.</i>	44
7.11	<i>Markov chains and distributions of \mathcal{A}, α and \mathcal{B} for one simulation with Algorithm 7. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{A}. (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α. (d) Distribution of α after burn-in. (e) Markov chain for \mathcal{B}. (f) Distribution of \mathcal{B} after burn-in.</i>	46
7.12	<i>Markov chains and distributions of \mathcal{A}, α and \mathcal{B} for one simulation with Algorithm 7. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. (a) Markov chain for \mathcal{A}. (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α. (d) Distribution of α after burn-in. (e) Markov chain for \mathcal{B}. (f) Distribution of \mathcal{B} after burn-in.</i>	47

C.1	(Simulation 2) Markov chains and distributions of \mathcal{A} , α and \mathcal{B} for one simulation with Algorithm 7. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{A} . (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in. (e) Markov chain for \mathcal{B} . (f) Distribution of \mathcal{B} after burn-in.	60
C.2	(Simulation 2) Markov chains and distributions of \mathcal{A} , α and \mathcal{B} for one simulation with Algorithm 7. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. (a) Markov chain for \mathcal{A} . (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in. (e) Markov chain for \mathcal{B} . (f) Distribution of \mathcal{B} after burn-in.	61

Contents

1	Introduction	1
1.1	Outline	2
2	Radar Principles	3
2.1	Basic Principles and Signal Processing	3
2.1.1	Radar	3
2.1.2	Digital Array Antennas	3
2.1.3	Pulse-Doppler Radar	4
2.1.4	Radar Range Equation and the Radar Cross Section	5
2.2	Sea Clutter Modeling	6
2.2.1	Signal Model	6
3	Elements of Stochastic Analysis	8
3.1	Stochastic Differential Equations	8
3.2	Stochastic Processes	8
3.2.1	Markov Process	8
3.2.2	Ornstein-Uhlenbeck Process	9
3.2.3	Cox-Ingersoll Ross Process	9
3.3	Numerical Methods	10
3.3.1	Euler-Maruyama Method	10
3.3.2	Milstein Method	10
4	Bayesian Inference	12
4.1	The Bayesian Framework	12
4.2	Metropolis-Hastings	12
4.2.1	Adaptive Metropolis-Hastings	14
4.3	Bayesian Filtering	15
4.3.1	State-Space Models	15
4.3.2	Particle Filters	17
5	Models for Reflectivity	20
5.1	K-Distribution Model	20
5.2	Random Walk Model	21
5.3	Field's Model	22
6	Methodology	23
6.1	Synthetic Time Series Data Ψ_t	23
6.2	Estimators for x_t and γ_t based on Ψ_t	24
6.3	Bayesian Estimation of $\mathcal{A}, \alpha, \mathcal{B}$ using Metropolis-Hastings	25
6.3.1	Estimation of \mathcal{B}	25

6.3.2	Estimation of A and α	26
6.4	Estimation of $\mathcal{A}, \alpha, \mathcal{B}$ using Bayesian Filtering and Metropolis-Hastings	28
6.4.1	State-Space Model	29
6.4.2	Particle Filter	29
6.4.3	Particle Filter within Metropolis-Hastings	31
6.4.4	Tuning the Particle Filter	33
7	Results	34
7.1	Estimators for x_t and γ_t	34
7.2	Bayesian Estimation Algorithms for Estimating Sea Clutter Parameters	37
7.2.1	Metropolis-Hastings for \mathcal{B}	37
7.2.2	Metropolis-Hastings for A and α	39
7.2.3	Selected N, T and σ for an Effective Particle Filter	42
7.2.4	Particle Filter within Metropolis-Hastings for $\mathcal{A}, \alpha, \mathcal{B}$	44
8	Future Work	49
9	Conclusion	50
	Bibliography	51
A	Algorithms	53
A.1	Sequential Importance Resampling	54
A.2	The Bootstrap Filter	55
B	Derivations	56
B.1	Estimator \bar{x}_t	56
B.2	Estimator \hat{x}_t	57
B.3	Estimator \tilde{x}_t	58
C	Figures	59

Introduction

Today, there is a wide range of radars, specialized to operate in different environments, such as sea, air and land. However, the purpose is usually the same; to detect targets. For example, weather radars are designed to detect precipitation and other atmospheric phenomena while military radars are used for surveillance and target tracking. In this thesis, radars operating in marine environments are considered. These radars encounter the same phenomena; *sea clutter*, which is unwanted reflections from the sea surface that greatly limit the radar's ability to detect slow, moving targets [2]. Hence, radar systems operating in marine environments must be able to manage the presence of sea clutter. In particular, this can be done by accurately describing the sea clutter, which can then be used to improve tracking and detection algorithms. This thesis aims to provide an accurate description of sea clutter using an approach based on the theory of *Stochastic Differential Equations* (SDE). SDEs are suitable due to their ability to capture the dynamic and stochastic nature of sea clutter.

A radar is an electromagnetic sensor for detection and tracking of moving objects [2]. From an antenna, the radar radiates electromagnetic energy that propagates in space. Eventually, some of this energy is intercepted by a reflecting object, the radar target, located at some distance from the radar. The intercepted energy is then reflected in many directions, where some of this energy is returned to and received by the radar antenna. In the case of maritime radar systems, a considerable part of this energy is attributed to sea clutter.

The statistical characteristics of sea clutter may vary widely. In order to develop suitable signal processing strategies for target detection and tracking, one needs to understand these characteristics. Therefore, there is a need for statistical models that accurately describe sea clutter [3]. An accurate description of sea clutter improves the radar performance and consequently enables target detection. For low sea state or land clutter the Gaussian distribution is a valid approximation, but in high sea state this is not the case anymore. For a considerable duration, the K-distribution model has served as the predominant statistical model for characterizing the backscattered signal from the sea surface [4]. However, this model is constrained by its inability to depict the dynamic nature of sea clutter. Indeed, the sea surface is not static, as it changes over time. Therefore, an accurate statistical model for sea clutter should incorporate the dynamics.

Fortunately, an appropriate dynamical model, referred to as *Field's model*, has been discovered by Timothy Field and it is thoroughly described in his monograph [4]. The model has been studied and clarified by Clement Roussel in his PhD thesis [1]. Field's model is a dynamical extension of the K-distribution model, where the dynamics in the model are expressed by stochastic differential equations, that include the roughness of the sea surface. The model is parameterized by three parameters: \mathcal{A} , \mathcal{B} and α , which depends on both the sea state and the characteristics of the emitted electromagnetic wave, including its frequency and polarisation [5].

As stated earlier, the goal is to find a statistical model that accurately describe the sea clutter. For this purpose, Field's model is considered. In particular, this thesis aims to estimate the unknown parameters \mathcal{A} , \mathcal{B} and α by using Bayesian estimation methods. Consequently, the research question can be formulated as:

Is it possible to derive an accurate and robust model for estimating the sea clutter parameters in Field's model with Bayesian inference?

All calculations and simulations in this thesis are performed using MATLAB.

1.1 Outline

In Chapter 2, basic principles of pertinent radar theory are introduced, focusing on the transmitted and received signals of monostatic radar systems. Chapter 3 presents fundamental concepts regarding stochastic analysis, beginning with the introduction of stochastic differential equations. This is followed by a description of the Ornstein-Uhlenbeck process and the Cox-Ingersoll-Ross process, both which are stochastic processes within Field's model. The chapter ends with a presentation of numerical methods for solving stochastic differential equations. Chapter 4 presents methods of Bayesian inference. Specifically, the Metropolis-Hastings method and Bayesian filtering methods are explained. In Chapter 5, statistical models for describing the complex reflectivity are introduced. Of particular significance is the description of Field's model, which forms the crux of this thesis. Chapter 6 presents the methods used for estimating the sea clutter parameters within Fields model. Particularly, two different approaches are applied; both employing the Metropolis-Hastings method but one that incorporates a particle filter for likelihood approximation. The results are then displayed in Chapter 7. Thereafter, Chapter 8 discusses the results and proposes directions for future work. Finally, in Chapter 9, the research question is answered and conclusions are drawn.

Radar Principles

Chapter 2 provides an overview of basic radar principles, signal processing and sea clutter modeling. In Section 2.1, the basic ideas of *digital array antennas* and *pulse-Doppler radar* are provided. Moreover, this section also describes the monostatic *radar range equation* and its connection to the *radar cross section*, that is a measure of the average power of the reflected echo. Furthermore, in Section 2.2, a brief presentation of signal modeling is provided. Understanding the concepts in this chapter are essential for effective radar system design and target detection in various environments. Statistical models for describing the complex reflectivity are then presented in Chapter 5.

2.1 Basic Principles and Signal Processing

2.1.1 Radar

Radar stands for RAdio Detection and Ranging [6]. It refers to electronic sensors designed for detecting and tracking objects at some distance from the radar. A radar transmits electromagnetic radiation that propagates in space. Radiation is intercepted by an object or target, located at some distance from the radar, and then returned to the radar as an echo. Based on the received signal, it is possible to make inference about the objects and targets.

As an electromagnetic wave propagates through the atmosphere, there is an interaction with water molecules leading to a reduction of intensity, which is called attenuation [6]. Since radio frequencies have relatively low attenuation, radars usually operate in that spectrum, with a large majority operating in the range of 300MHz to 35GHz.

A radar system can have multiple transmitting and receiving antennas, but the most common configuration, has one antenna that switches between transmitting and receiving (recording) [7]. The former is called a *multistatic radar* and the latter a *monostatic radar*. A monostatic radar is blind while it transmits.

2.1.2 Digital Array Antennas

Historically, antennas were parabolas that directed the radiation by means of the shape of the antenna. Modern radars are so called array antennas, which are built as an array of antenna elements [7]. Each antenna element is a simple antenna that transmits and receives electromagnetic waves without any significant beam shaping or directional control. However, when the antenna elements in an array antenna are working together, interference creates a directivity of the energy. By arranging the antenna elements in specific geometric patterns, desired radiation is received. The more elements there are in each dimension, the stronger is the directivity and most energy is concentrated in a narrow beam.

In a fully digital antenna there is an analogue to digital converter behind every antenna element, creating a huge amount of digital data [6]. Digital beamforming, is the process of summing the weighted contributions of individual elements (in software). In transmission, the individual signals are not directed, but the sum is. Changing phases in the signal before summing (again in software),

creates directivity. Often, in modern systems, a wide transmission beam is used (as it cannot be changed in the processing stage) and many receive directions are used as it can be created digitally in the signal processing.

2.1.3 Pulse-Doppler Radar

A pulse-Doppler radar is a radar system that uses *pulse timing techniques* to determine the range to a target, and uses the *Doppler effect* of the returned signal for estimating the velocity of the target. The key advantage of pulse-Doppler radar is its ability to detect small moving targets in severe clutter environments [2]. A pulse-Doppler radar emits electromagnetic pulses with a fixed *Pulse Repetition Frequency* (PRF) and receives and records signals between the pulses. In this section monostatic radars are considered.

The Range from the radar to the target can be determined by using pulse delay ranging [8]. Thus, the range R is given by

$$R = \frac{\tau c}{2},$$

where τ denotes the time delay, which is the time between the emitted pulse and the received echo from the target, and c is the speed of light.

The radial velocity between the radar and the target v is assessed through the Doppler effect, which is a phase shift of the complex signal, pulse by pulse [2]. It is given by

$$v = -\frac{f_d \lambda}{2}.$$

Here, f_d is the Doppler frequency shift, $\lambda = c/f_c$ is the carrier wavelength, c is again the speed of light and f_c is the carrier frequency. When the target is moving away from the radar, the radial velocity is positive and the Doppler effect is negative. Contrary, when the target is moving towards the radar, the radial velocity is negative and the Doppler effect is positive.

Ambiguities arise in both range and velocity estimations. For range, ambiguities arise from the fact that a multitude of pulses are transmitted and matching them with the received pulses is not possible. For velocity, they stem from aliasing and the finite number of pulses used in the processing.

Before describing the sea clutter, which is the core of this thesis, the data from a digital array antenna is presented more carefully. The transmitted pulse from one element can be written on the form

$$S_{\text{transmit}}(t) = e^{i2\pi f_c t} E(t) = [\cos(2\pi f_c t) + i \sin(2\pi f_c t)] E(t).$$

Here, $E(t)$ is the so called envelope that defines the transmitted pulse and f_c is the carrier frequency [6]. A typical envelope is the linear chirp $E(t) = e^{jat^2} \mathbf{1}_{[0,t_0]}(t)$, where t_0 determines the duration of the pulse and a is a constant. A tiny reflection of the transmitted signal is returned to the antenna and recorded. The received signal can approximately be written as

$$S_{\text{received}}(t) = \Psi(t) S_{\text{transmit}}(t - \tau),$$

where τ is the time delay and $\Psi(t)$ is the complex coefficient which accounts for amplitude and phase changes of the transmitted signal [1]. One antenna element can send multiple pulses. An antenna element that transmits multiple pulses gives rise to a complex matrix with two different time dimensions. The dimension corresponding to the time delay of the pulses is referred to as *fast time* and the dimension corresponding to consecutive pulses is called *slow time*. If neither the target nor the radar are moving, the time delay τ remains the same between each inter pulse recording interval. However, when there is a very high radial velocity the time delay will change significantly, resulting in the failure of classical signal processing. For small or moderate radial velocities towards or from the radar, and due to the short time scales in fast and slow time, the time delay is still approximately constant. However, the Doppler effect gives for constant velocities rise to a linear phase shift, pulse by pulse. With a more precise notation, the digital signal from a point shaped target at range R with

radial velocity v , measured with N pulses from one antenna element, has the form of the rank one matrix

$$S_{\text{digital}} = \mathbf{a}(f_d) \otimes \mathbf{b}(\tau).$$

Here the so called steering vectors \mathbf{a} and \mathbf{b} are given by

$$\begin{aligned} \mathbf{a}(f_d) &= (1, e^{1i \cdot 2\pi f_d / f_s}, e^{2i \cdot 2\pi f_d / f_s}, \dots, e^{(N-1)i \cdot 2\pi f_d / f_s}), \\ \mathbf{b}(\tau) &= (E(s_0 - \tau), E(s_1 - \tau), E(s_2 - \tau), \dots, E(s_M - \tau)), \end{aligned}$$

where f_d is the Doppler shift and f_s is the pulse repetition frequency. If one considers the signals from several antenna elements or channels, the matrix is stacked in a 3-tensor called the radar data cube. This third dimension is called the *spatial dimension* as the elements are distributed in space and digital beamforming give directional information. However, the focus remains on the consideration of one element.

To increase the probability of detection and decrease the probability of false alarm, methods to amplify the signals relative to the noise are employed. This can be achieved by a matched filter, also called *pulse compression*. The technique compresses the wide pulse into a narrower spike by cross correlation with the transmitted signal. In classical signal processing, pulse compression is followed by *Doppler filtering*, which enables localization of objects with velocities. This methodology makes it possible to infer τ and thus R .

2.1.4 Radar Range Equation and the Radar Cross Section

In Section 2.1.3, the signal model for one target was described up to a complex amplitude. The magnitude of this complex amplitude is determined by the so called radar range equation, which is introduced in this section. The radar range equation connects target properties with radar characteristics, the distance between the radar and the target, and the properties of the medium [9]. There are several forms of the radar range equation [6]. Importantly, the radar range equation for a monostatic radar can be formulated as

$$P_{\text{receive}} = \frac{P_{\text{transmit}} A^2 f_c^2}{4\pi L R^4 c^2} \sigma. \quad (2.1)$$

The quantities in the formula are next listed.

- P_{receive} : Peak received power in Watts.
- P_{transmit} : Peak transmitted power in Watts.
- A : Effective antenna area.
- f_c : Carrier frequency.
- c : Speed of light.
- σ : Radar Cross Section (RCS), which measures the average power of the reflected echo.
- L : Losses due to imperfections and thermal noise.
- R : Range from the radar to the target in meters.

In practice, there is always a loss L , which is why it is chosen to be included in Equation 2.1. In idealized form, the loss is excluded. The loss L is a combination of several losses, in particular atmospheric attenuation (absorption in water molecules in the atmosphere), that is range dependent.

To describe the RCS of extended targets, for example the sea surface, it is convenient to define the radar cross section per unit area of a illuminated surface

$$\sigma_0 = \frac{\sigma}{A}.$$

Here σ is again the RCS, A is the area of the illuminated sea surface and σ_0 is the normalised RCS [6]. It is a well-known fact that σ_0 depends on incident angle, polarization, and transmitter wavelength. Depending on the specific nature of the illuminated surface, σ_0 can also depend on factors, such as, wind speed and direction, level of moisture, and surface roughness.

2.2 Sea Clutter Modeling

As discussed in Section 2.1.3, radar data has three dimensions, fast time (range), slow time (pulses) and channel (antenna elements). This thesis only considers one of the dimensions, namely slow time. First, it is assumed that the data stems from a linear combination of channels after digital beamforming, which reduces the 3-tensor to a matrix. Furthermore, the data is assumed to be pulse compressed and a fixed range is considered, where the primary scattering stems from a rough sea surface. This reduces the matrix to a complex vector in slow time.

If a radar emits electromagnetic signals towards a sea surface, the signals are reflected and some of the energy is sent back to the radar receiver [1]. In addition to containing information about the target, the backscattered signal also carries information about the reflections from the sea surface. This (often unwanted) backscatter from the sea surface is commonly referred to as sea clutter. The statistical nature of the sea clutter depends on many factors, for example wind, sea waves (gravitational and capillary) and surface currents [10]. Sea clutter statistics also depends heavily on various radar system parameters, such as carrier frequency, polarisation, and the bandwidth of the waveform. To account for all these factors makes sea clutter modeling a very difficult task. In this thesis, the focus is on describing the scattering for a *fixed geometry*.

2.2.1 Signal Model

In order to formulate an accurate model for describing sea clutter, it is important to understand the underlying electromagnetic aspects based on physical principles, that give rise to the observed clutter signal. Consequently, adequate clutter signals can be generated. An immediate application of these generated signals is to enhance the evaluation of receiver algorithms such as target detection algorithms, especially in situations where the amount of real data is insufficient. The objective of this thesis is to view sea clutter using the framework of stochastic differential equations, employing an approach based on first-principles assumptions.

A fundamental part in sea clutter modeling is to accurately include its time characteristics. Although the literature on sea clutter modeling is extensive, the explicit inclusion of time is typically neglected. Hence, there is a need to model sea clutter where time is explicitly included. This should be compared with an approach where the sea clutter statistics is considered without any time dependence. This thesis aims to view sea clutter as formally arising from solutions to the stochastic differential equations constituting Field's model, which will be described in Section 5.3. A consequence of this approach is that the sea clutter statistics can be parameterised in terms of parameters with a clear physical meaning.

Assume the presence of a target at the range of consideration. The reflected signal at this range can be written as

$$\mathbf{y} = \mathbf{s} + \mathbf{w} + \mathbf{c}.$$

Here, \mathbf{s} is the signal from the target, \mathbf{w} is the complex Gaussian noise, and \mathbf{c} denotes the clutter from land or sea. It is learned that the response in slow time from a target with velocity v is given by $\mathbf{s} = \alpha \mathbf{a}(f_d)$, where f_d is the Doppler frequency shift related to v , $|\alpha|$ is determined by the radar range equation and the phase of α is essentially non-informative. The noise \mathbf{w} stems from thermal noise and is well modelled as Gaussian. Since the clutter originates from land or sea, and thus a continuous extended surface, it becomes meaningful to approximate the response with a multitude of point scatterers. The contribution from each scatterer is independent and from the central limit theorem it is known that the sum of many independent contributions is almost Gaussian. For land and

calm sea, this is a very good approximation and the Gaussian assumption is employed to detect targets in a statistical hypothesis test. However, for high sea with big and braking waves this assumption is no longer well satisfied, since waves are extended and this violates the independence of individual scatterers.

Assuming that the clutter originates from N independent scatterers with amplitudes $\alpha_1, \dots, \alpha_N$ and velocities v_1, \dots, v_N , the clutter signal reads

$$\mathbf{c} = \sum_{n=1}^N \alpha_n \mathbf{a}_n(f_{d,n}),$$

where the vector \mathbf{c} is considered a complex valued function in slow time t . By using the notation from the work by Roussel [1], the clutter signal, when a radar illuminates a fixed portion of the sea with some fixed PRF, is a discrete time series of the complex reflectivity denoted as

$$\Psi_{t_k} = R_{t_k} + iI_{t_k}, \quad k = 1, \dots, n, \quad (2.2)$$

where $t_k - t_{k-1} = 1/PRF$ for all k , and R and I is the real and imaginary part of the reflectivity, respectively.

Elements of Stochastic Analysis

In this chapter fundamental concepts regarding stochastic analysis that are of importance for this thesis are defined and explained. Firstly, the concept of stochastic differential equations is introduced in Section 3.1. This is followed by a presentation of relevant stochastic processes in Section 3.2. Importantly, the *Ornstein-Uhlenbeck* (OU) process and the *Cox-Ingersoll-Ross* (CIR) process are introduced. To solve SDEs analytically is in most cases impossible. Thus, numerical methods need to be implemented. In Section 3.3, both the *Euler-Maruyama* and *Milstein* schemes are defined.

3.1 Stochastic Differential Equations

A stochastic differential equation is a differential equation in which one or more of the terms is a stochastic process [11, 12]. The stochastic process S_t with initial condition S_0 is a solution to the following SDE

$$dS_t = \mu(S_t)dt + \sigma(S_t)dW_t \quad 0 \leq t \leq T. \quad (3.1)$$

Here, $(W_t)_{t \in [0, T]}$ is a Brownian motion or Wiener process and the functions μ and σ are called the drift and volatility, respectively. Equation (3.1) is a convenient notation for the following stochastic equation

$$S_t = S_0 + \int_0^t \mu(S_u)du + \int_0^t \sigma(S_u)dW_u \quad 0 \leq t \leq T.$$

The rightmost integral is known as the *Itô stochastic integral* and was first defined by Kiyosi Itô in 1949.

SDEs are used to model systems that experience random behaviour over time. Hence, they have applications in many areas, notable examples being financial mathematics and physics. For instance, they can be used for describing interest rates and stock prices. In this thesis, stochastic differential equations describe the complex reflectivity of the sea surface.

3.2 Stochastic Processes

This section presents stochastic processes that are crucial for the comprehension of Fields's model for reflectivity, as detailed in Section 5.3. Specifically, Fields's model incorporates two significant Markov processes: the Cox-Ingersoll-Ross process and the Ornstein-Uhlenbeck process. For a thorough understanding of Field's model, this section first introduces the Markov process in Section 3.2.1. This is followed by a presentation of the Ornstein-Uhlenbeck process and the Cox-Ingersoll-Ross process in Section 3.2.2 and Section 3.2.3, respectively.

3.2.1 Markov Process

A stochastic process that satisfies the so called *Markov property* is called a Markov process [13, 14]. According to the Markov property, the future state of the system depends only on the present

state, and not on the past states. There is only a memory from the random variable at the current time. Therefore, for a Markov process the conditional probability distribution only depends on the value at the current time. The stochastic process S_t is a continuous Markov process if $\forall k \in \mathbb{N}$, and $\forall t_1 < t_2 < \dots < t_k$, the following holds true [1]

$$p(S_{t_k} = s_k | S_{t_1} = s_1, \dots, S_{t_{k-1}} = s_{k-1}) = p(S_{t_k} = s_k | S_{t_{k-1}} = s_{k-1}).$$

3.2.2 Ornstein-Uhlenbeck Process

An Ornstein-Uhlenbeck process is a stochastic process with applications in many fields, such as financial mathematics and physics [15, 16]. The OU process is a stationary, Markovian, Gaussian process, in fact the only process with all these properties [15]. A stationary process has statistical properties that remains constant over time, and a Gaussian process is a collection of random variables such that any finite subset of it has a joint multivariate Gaussian distribution. Over time, there is a drift towards the mean function μ for the process. The Ornstein-Uhlenbeck process S_t is a solution to the following SDE

$$dS_t = \theta(\mu - S_t)dt + \sigma dW_t.$$

Here, the parameter $\theta > 0$ corresponds to the speed of adjustment to the mean μ and $\sigma > 0$ represents the volatility.

A sample path of the OU process, starting at $S_0 = 2$, is shown in Figure 3.1. The trajectory is generated by the Euler-Maruyama method, presented in Section 3.3.1. The parameters are $\mu = 0.5$, $\theta = 0.1$ and $\sigma = 0.1$.

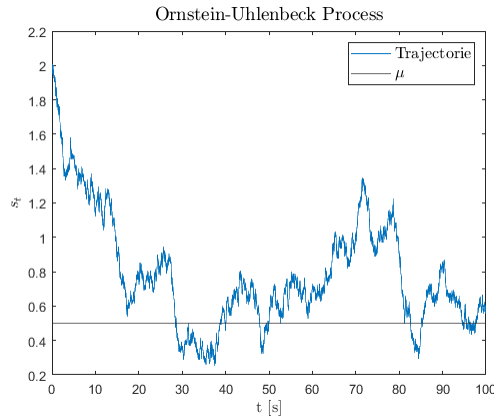


Figure 3.1: *Sample path of an Ornstein-Uhlenbeck process, with $\mu = 0.5$, $\theta = 0.1$, $\sigma = 0.1$ and initial value $S_0 = 2$. The black line corresponds to the long term mean μ .*

3.2.3 Cox-Ingersoll Ross Process

Similarly to the OU process, the Cox-Ingersoll-Ross process is a stochastic process with applications in many areas [17]. The CIR process S_t is a stationary Markov process, defined by the following SDE

$$dS_t = k(\theta - S_t)dt + \sigma\sqrt{S_t}dW_t.$$

Here $k > 0$, $\theta > 0$ and $\sigma > 0$ are constants. The CIR model describes how the process S_t is evolving over time, where the parameter θ corresponds to the mean of the process, the parameter k corresponds to the speed of adjustment to the mean θ and σ represents the volatility. Moreover, to ensure that the process S_t remains positive over time, the following condition has to be imposed

$$2k\theta > \sigma^2.$$

A sample path of the CIR process, starting at $S_0 = 2$, is shown in Figure 3.2. The trajectory is generated by the Euler-Maruyama method, presented in Section 3.3.1. The parameters are $\theta = 0.5$, $k = 0.1$ and $\sigma = 0.1$.

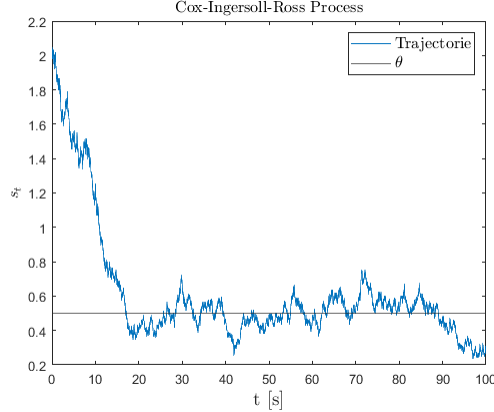


Figure 3.2: *Sample path of a Cox-Ingersoll-Ross process, with $\theta = 0.5$, $k = 0.1$, $\sigma = 0.1$ and initial value $S_0 = 2$. The black line corresponds to the long term mean θ .*

3.3 Numerical Methods

For most SDEs, there are no closed-form analytical solutions. Consequently, one needs to solve the stochastic differential equations by instead using numerical methods. Two such methods are the Euler-Maruyama and Milstein schemes, which are described in Section 3.3.1 and 3.3.2, respectively. The main idea is to approximate the solution to a stochastic differential equation by discretizing the SDE over small time steps and recursively updating the solution in time [11]. The Euler-Maruyama and Milstein schemes are considered since these are the methods employed by Roussel in [1] for discretization of Field’s model.

3.3.1 Euler-Maruyama Method

To apply the Euler-Maruyama method to the SDE (3.1) over the duration $[0, T]$, the interval is first discretized [12]. Let $\Delta t = T/N$ for some positive integer N , and $t_k = k\Delta t$. Let S_k denote the numerical approximation to S_{t_k} . Thus, the Euler-Maruyama scheme becomes

$$S_k = S_{k-1} + \mu(S_{k-1})\Delta t + \sigma(S_{k-1})\Delta W_k, \quad k = 1, 2, \dots, N, \quad (3.2)$$

with the initial value $S_{t_0} = S_0$. $\Delta W_k = W_k - W_{k-1}$ is a Brownian increment with distribution $\Delta W_k \sim \mathcal{N}(0, \Delta t)$.

3.3.2 Milstein Method

The Milstein method is the same as Euler-Maruyama except that it has a correction term, which provides it with better convergence properties than the Euler-Maruyama method [12]. Consider the SDE (3.1). As before, let $\Delta t = T/N$ for some positive integer N , and $t_k = k\Delta t$. Let S_k denote the numerical approximation to S_{t_k} . Thus, the Milstein scheme becomes

$$S_k = S_{k-1} + \mu(S_{k-1})\Delta t + \sigma(S_{k-1})\Delta W_k + \frac{1}{2}\sigma(S_{k-1})\frac{d\sigma}{dS}(S_{k-1})((\Delta W_k)^2 - \Delta t), \quad k = 1, 2, \dots, N, \quad (3.3)$$

with the initial value $S_{t_0} = S_0$. Again, $\Delta W_k = W_k - W_{k-1}$ is a Brownian increment with distribution $\Delta W_k \sim \mathcal{N}(0, \Delta t)$. This formulation of the Milstein scheme requires that the SDE is one-dimensional, otherwise the correction term becomes way more complicated and not practically computable [18]. The Euler-Mauyama scheme directly applies in any dimension.

4

Bayesian Inference

Bayesian inference techniques are methods of statistical inference in which *Bayes' Theorem* constitutes an essential part. In Chapter 4 the relevant theory of Bayesian inference is presented. The chapter begins with Section 4.1, which gives an introduction to the Bayesian Framework, including Bayes' Theorem. Section 4.2 describes the *Metropolis-Hastings* (MH) method, which is a powerful algorithm for the approximation of posterior distributions. Lastly, in Section 4.3, elements of *Bayesian filtering* methods are presented, including state-space models and particle filtering methods.

4.1 The Bayesian Framework

Let θ denote parameters to be estimated and let \mathbf{x} be data which contains information about the parameters θ . A Bayesian estimation of θ is based on Bayes' Theorem

$$p(\theta|\mathbf{x}) = \frac{\mathcal{L}(\theta|\mathbf{x})\pi(\theta)}{p(\mathbf{x})} \propto \mathcal{L}(\theta|\mathbf{x})\pi(\theta), \quad (4.1)$$

where $p(\theta|\mathbf{x})$ is the *posterior* distribution of θ , $\mathcal{L}(\theta|\mathbf{x}) = p(\mathbf{x}|\theta)$ is the *likelihood*, $\pi(\theta) = p(\theta)$ is the *prior* distribution of θ and $p(\mathbf{x})$ is the *marginal likelihood* [1].

In many Bayesian estimation problems the likelihood $\mathcal{L}(\theta|\mathbf{x})$ does not have a closed-form expression. In such case it is either impossible or very complicated to analytically compute the posterior distribution $p(\theta|\mathbf{x})$. Instead, numerical methods can be used to numerically estimate the posterior distribution. Today, the most frequently used methods for simulating complicated or high dimensional distributions are *Markov Chain Monte Carlo* (MCMC) methods. The main idea is to generate a Markov chain that converges to the posterior distribution, which is called the *target distribution* [19]. More precisely, the stationary distribution of the Markov chain, to which the Markov chain converges, should coincide with the posterior distribution. There are several types of MCMC algorithms, for example Metropolis-Hastings, which is described in the next section.

4.2 Metropolis-Hastings

The aim of this section is to provide a detailed explanation of the Metropolis-Hastings method. Consider the problem of estimating the unknown parameters θ given some observed data \mathbf{x} . As demonstrated by Bayes' Theorem (4.1), the posterior is proportional to the likelihood times the prior

$$p(\theta|\mathbf{x}) \propto \mathcal{L}(\theta|\mathbf{x})\pi(\theta).$$

When working with likelihood functions, it is often convenient to consider the logarithm of the likelihood function $\ell(\theta|\mathbf{x}) = \log \mathcal{L}(\theta|\mathbf{x})$, which is referred to as the *log-likelihood function*. The reason why this is preferred is because of the large range of the values constituting the likelihood function, which sometimes leads to computational difficulties when applying the problem to some programming language, for instance MATLAB. More precisely, the likelihood usually factorizes into many terms,

resulting in a wide range of possible values. However, when taking the logarithm of the likelihood these factors become a sum, effectively reducing the range of possible values.

The Metropolis-Hastings algorithm is presented in Algorithm 1. The method starts by initializing a value θ_0 , which is either drawn from the prior distribution $\pi(\theta)$ or chosen as a reasonable value. At every iteration a new value θ^* is sampled from a *proposal distribution* $q(\theta^*|\theta_{i-1})$. The candidate is either accepted or rejected depending on how likely it is to belong to the target distribution. The *acceptance probability* consists of two relations and is given by

$$\alpha = \min \left\{ 1, \frac{\mathcal{L}(\theta^*|\mathbf{x})\pi(\theta^*)}{\mathcal{L}(\theta_{i-1}|\mathbf{x})\pi(\theta_{i-1})} \frac{q(\theta_{i-1}|\theta^*)}{q(\theta^*|\theta_{i-1})} \right\}.$$

The first factor is the ratio between the posterior distribution of the candidate θ^* and the previous value θ_{i-1} . To show why the normalizing distribution $p(\mathbf{x})$ is not needed, one can write

$$\frac{p(\theta^*|\mathbf{x})}{p(\theta_{i-1}|\mathbf{x})} = \frac{\mathcal{L}(\theta^*|\mathbf{x})\pi(\theta^*)}{p(\mathbf{x})} \bigg/ \frac{\mathcal{L}(\theta_{i-1}|\mathbf{x})\pi(\theta_{i-1})}{p(\mathbf{x})} = \frac{\mathcal{L}(\theta^*|\mathbf{x})\pi(\theta^*)}{\mathcal{L}(\theta_{i-1}|\mathbf{x})\pi(\theta_{i-1})}.$$

The second factor is the ratio between the proposal distribution of the previous value given the candidate and the proposal distribution of the candidate given the previous value. As a result, the proposal is punished if it moves to regions from where it is more difficult to come back from. If the proposal distribution is symmetric, $q(\theta^*|\theta_{i-1}) = q(\theta_{i-1}|\theta^*)$, then the second factor in the acceptance probability can be removed since it is always equal to 1.

Next, it is evaluated whether the proposal θ^* should be accepted or not. A random variable u is drawn from the uniform distribution $\mathcal{U}(0, 1)$. The proposal θ^* is accepted if $u \leq \alpha$, and the next value in the algorithm becomes $\theta_i = \theta^*$. If instead $u > \alpha$, the proposal is rejected, and the previous value remains $\theta_i = \theta_{i-1}$.

In situations where it is more convenient to consider the log-likelihood function, the acceptance probability is calculated according to

$$\log \alpha = \min \{ 0, \ell(\theta^*|\mathbf{x}) - \ell(\theta_{i-1}|\mathbf{x}) + \log \pi(\theta^*) - \log \pi(\theta_{i-1}) + \log q(\theta_{i-1}|\theta^*) - \log q(\theta^*|\theta_{i-1}) \}.$$

The recently described steps are repeated for a sufficient number of iterations N . The number of iterations needed, depends on how fast the Markov chain converges to the target distribution $p(\theta|\mathbf{x})$. Importantly, the performance of the Metropolis-Hastings algorithm depends on the choice of the proposal distribution q [20]. However, as $N \rightarrow \infty$ the distribution of the state θ_i at iteration i converges to the target distribution. The time it takes for the Markov chain to converge sufficiently close to the target distribution is called the *burn-in period*. If all the samples in the burn-in period are discarded, then the Markov chain constitutes an empirical distribution that can be considered an estimation of the target distribution.

Choosing an appropriate proposal distribution q is crucial for the performance of the MH algorithm [21]. To find an appropriate proposal distribution can be difficult. If the proposal distribution explores too widely, then the Markov chain might take a long time to converge, leading to slow mixing and high autocorrelation between samples. On the other hand, if the proposal distribution is too narrow, then the Markov chain might get stuck in a local region of the distribution and fail to explore the entire space. The next Section describes an *adaptive proposal* that aims to address this quandary.

Algorithm 1 The Metropolis-Hastings Algorithm

Input: \mathbf{x}, N **Output:** $\theta_{1:N}$ **1. Initialization**(a) Draw θ_0 from the prior distribution $\pi(\theta_0)$.**2. Repeat for $i = 1 : N$** **1. Proposal**(a) Sample a candidate θ^* from the proposal distribution $q(\theta^*|\theta_{i-1})$.(b) Calculate the likelihoods: $\mathcal{L}(\theta^*|\mathbf{x})$ and $\mathcal{L}(\theta_{i-1}|\mathbf{x})$.(c) Calculate the acceptance probability: $\alpha = \min \left\{ 1, \frac{\mathcal{L}(\theta^*|\mathbf{x})\pi(\theta^*)}{\mathcal{L}(\theta_{i-1}|\mathbf{x})\pi(\theta_{i-1})} \frac{q(\theta_{i-1}|\theta^*)}{q(\theta^*|\theta_{i-1})} \right\}$.**2. Acceptance or Rejection**(a) Draw $u \sim \mathcal{U}(0, 1)$.(b) If $u \leq \alpha$, accept θ^* and set $\theta_i = \theta^*$.(c) If $u > \alpha$, reject θ^* and set $\theta_i = \theta_{i-1}$.

4.2.1 Adaptive Metropolis-Hastings

The difficulties of tuning the proposal distribution in the MH algorithm can be avoided by introducing an adaptive proposal [21]. The idea is to update the proposal distribution based on the knowledge learned about the target distribution so far. Assume that the target distribution of interest is the same as in the previous section, namely $p(\theta|\mathbf{x})$. The adaptive proposal is a Gaussian distribution with mean at the current value θ_i and covariance $C_i = C_i(\theta_0, \dots, \theta_i)$ according to [22]

$$q_{C_i}(\theta^*|\theta_i) \sim \mathcal{N}(\theta^*|\theta_i, C_i).$$

Note that the proposal distribution is symmetric and hence it can be removed from the calculation of the acceptance probability.

The *Adaptive Metropolis-Hastings* (AMH) method, presented in Algorithm 2, starts by initializing the parameter θ_0 , and unlike the ordinary MH algorithm also the covariance C_0 . Additionally, a starting time t_A at which to start the adaptive process is chosen. The initial covariance is kept for the first t_A iterations. Then, at time t_A a new covariance matrix C_i is computed based on the chain of accepted proposals up to that point. The covariance matrix is given by the following equations

$$C_i = \begin{cases} C_0, & i < t_A, \\ s_d \text{cov}(\theta_0, \dots, \theta_i) + s_d \varepsilon I_d, & i \geq t_A, \end{cases}$$

where s_d is a constant that depends on the dimension d of the parameter space. A basic choice of s_d is given by $s_d = (2.4)^2/d$ [21]. To avoid numerical instability a small diagonal matrix $s_d \varepsilon I_d$, where $0 < \varepsilon \ll 1$, is added [22]. Moreover, the sample covariance is calculated according to

$$\text{cov}(\theta_0, \dots, \theta_i) = \frac{1}{i-1} \sum_{k=0}^i (\theta_k - \bar{\theta}_i)^T (\theta_k - \bar{\theta}_i),$$

where $\bar{\theta}_i = \frac{1}{i+1} \sum_{k=0}^i \theta_k$ is the sample mean of the chain up to that point. At each subsequent time step when $i \geq t_A$ the covariance is updated recursively according to the following equations

$$\bar{\theta}_i = \frac{1}{i+1} \sum_{k=0}^i \theta_k, \quad (4.2)$$

$$C_i = \frac{i-1}{i} C_{i-1} + \frac{s_d}{i} (i\bar{\theta}_{i-1}\bar{\theta}_{i-1}^T - (i+1)\bar{\theta}_i\bar{\theta}_i^T + \theta_i\theta_i^T + \varepsilon I_d). \quad (4.3)$$

Algorithm 2 The Adaptive Metropolis-Hastings Algorithm

Input: \mathbf{x} , C_0 , t_A , s_d , ε , N

Output: $\theta_{1:N}$

1. Initialization

(a) Draw θ_0 from the prior distribution $\pi(\theta_0)$.

2. Repeat for $i = 1 : N$

1. Covariance

(a) If $i < t_A$, set $C_i = C_{i-1}$.

(b) If $i = t_A$, compute $C_i = s_d \text{cov}(\theta_0, \dots, \theta_{i-1}) + s_d \varepsilon I_d$.

(c) If $i > t_A$, compute C_i recursively according to (4.2) and (4.3).

2. Proposal

(a) Sample a candidate θ^* from the proposal distribution $q_{C_i}(\theta^* | \theta_{i-1})$.

(b) Calculate the likelihoods: $\mathcal{L}(\theta^* | \mathbf{x})$ and $\mathcal{L}(\theta_{i-1} | \mathbf{x})$.

(c) Calculate the acceptance probability: $\alpha = \min \left\{ 1, \frac{\mathcal{L}(\theta^* | \mathbf{x}) \pi(\theta^*)}{\mathcal{L}(\theta_{i-1} | \mathbf{x}) \pi(\theta_{i-1})} \right\}$.

3. Acceptance or Rejection

(a) Draw $u \sim \mathcal{U}(0, 1)$.

(b) If $u \leq \alpha$, accept θ^* and set $\theta_i = \theta^*$.

(c) If $u > \alpha$, reject θ^* and set $\theta_i = \theta_{i-1}$.

4.3 Bayesian Filtering

Bayesian filtering methods, such as *Kalman filters* and *Particle Filters* (PF) are techniques that are frequently used for recursive state estimation problems [23]. This section begins with introducing relevant theory of state-space models in Section 4.3.1. Then, particle filters, also known as *Sequential Monte Carlo* (SMC) methods are presented in Section 4.3.2. These methods can be used to recursively estimate desired distributions when the state-space model is nonlinear.

4.3.1 State-Space Models

First, let $\{\mathbf{x}_k\}_{k=0}^T$ and $\{\mathbf{z}_k\}_{k=1}^T$ denote the *hidden states* and the *measured states* up to the final time step T , respectively. State-space models are dynamical systems satisfying the Markovian property, which means that the current state \mathbf{x}_k only depends on the previous state \mathbf{x}_{k-1} . Hence, a general state-space model can be described by the distributions [23]

$$\begin{aligned} \text{Initial distribution:} & \quad p(\mathbf{x}_0), \\ \text{Transition model:} & \quad p(\mathbf{x}_k | \mathbf{x}_{k-1}), \quad k = 1, \dots, T, \\ \text{Measurement model:} & \quad p(\mathbf{z}_k | \mathbf{x}_k), \quad k = 1, \dots, T. \end{aligned}$$

The *transition model* describes how the state \mathbf{x}_k is evolving over time and the *measurement model* relates measurements \mathbf{z}_k to the state \mathbf{x}_k . Figure 4.1 illustrates the relation between the hidden and measured states in terms of a Bayesian network. In the filtering of state-space systems, the objective is typically to compute or estimate the posterior distribution $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ recursively in time via Bayesian filtering methods [23]. That is, the goal is to estimate the hidden state sequence \mathbf{x}_k of a dynamical system given observations \mathbf{z}_k for the discrete times $k = 1, 2, \dots, T$.

In general, state-space models depend on some parameters θ . However, the conditioning on θ is chosen to not be written explicitly since these parameters can be considered to be included in the functions \mathbf{f}_k and \mathbf{h}_k below. The recurrence relations of the dynamical system can be expressed as follows

$$\begin{aligned}\mathbf{x}_k &= \mathbf{f}_k(\mathbf{x}_{k-1}) + \mathbf{q}_{k-1}, \\ \mathbf{z}_k &= \mathbf{h}_k(\mathbf{x}_k) + \mathbf{r}_k,\end{aligned}$$

where \mathbf{f}_k is a *transition function* that describes how the state \mathbf{x}_k evolves from time $k-1$ to k and \mathbf{h}_k is a *measurement function* that associates the state \mathbf{x}_k with a measurement \mathbf{z}_k . Moreover, $\mathbf{q}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ and $\mathbf{r}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$ are zero-mean multivariate Gaussian random variables with covariance matrices \mathbf{Q}_{k-1} and \mathbf{R}_k , respectively. If either \mathbf{f} or \mathbf{h} is nonlinear, then the system is considered a *nonlinear dynamical system* and if both \mathbf{h} and \mathbf{f} are linear, then the system is considered a *linear dynamical system*.

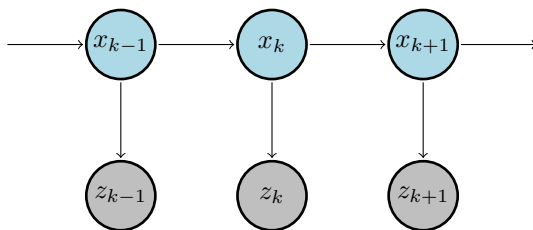


Figure 4.1: *Hidden Markov Model for a recursive state estimation problem.*

As stated earlier, Bayesian filtering methods aim to estimate the posterior state. By using Bayes theorem, a recursive formula for the posterior state $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ satisfies

$$\text{Prediction: } p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1}, \quad (4.4)$$

$$\text{Updating: } p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{\int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})d\mathbf{x}_k}. \quad (4.5)$$

Although the usual objective when dealing with state-space systems is to compute or estimate the posterior state, there are situations when the distribution of interest is instead the marginal distribution $p(\mathbf{z}_{1:T})$. This is highly relevant if the purpose is to do Bayesian inference with MH, where the goal is to determine the posterior distribution $p(\theta|\mathbf{z}_{1:T})$. When applying the MH algorithm, the marginal distribution $p(\mathbf{z}_{1:T})$ is conditioned on θ , resulting in the likelihood function $p(\mathbf{z}_{1:T}|\theta)$. The likelihood function can be obtained by filtering methods [24]. As a first step, this distribution is decomposed as

$$p(\mathbf{z}_{1:T}) = \prod_{k=1}^T p(\mathbf{z}_k|\mathbf{z}_{1:k-1}), \quad (4.6)$$

where each contribution to the product can be computed by

$$p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})d\mathbf{x}_k. \quad (4.7)$$

In the linear case, there exist a closed form solution to this distribution, namely the Kalman filter. Otherwise filtering methods, such as particle filters can be applied.

4.3.2 Particle Filters

Particle filters, also known as sequential Monte Carlo methods, can be used for computing the conditional distributions defined by Equations (4.4)-(4.7). SMC methods are computationally demanding but attractive in the sense that they enable estimation of distributions in nonlinear systems. In addition, these methods are very flexible, easy to implement and applicable in very general settings [23]. Particle filters can be adjusted to estimate the desired distributions.

The key idea with SMC methods is to generate new samples of the hidden states and weight these samples based on how close to the observations $\mathbf{z}_{1:T}$ they are. Consequently, Monte Carlo approximations can be produced for the conditional distributions given by Equations (4.4)-(4.7). Before outlining sequential Monte Carlo methods, there is a need in describing *importance sampling*, since SMC methods are an extension of this method designed for dynamic state-space models.

Importance sampling is a Monte Carlo method used to estimate the expected value of a function $\mathbf{g}(\mathbf{x})$ when direct sampling from the posterior distribution $p(\mathbf{x}|\mathbf{z})$ is difficult or inefficient. Instead, the method samples from another distribution called *importance distribution* $q(\mathbf{x}|\mathbf{z})$. Then, the samples are weighted to correct for the use of this distribution. This yields the following Monte Carlo approximation [23]

$$\begin{aligned}\mathbb{E}[\mathbf{g}(\mathbf{x})|\mathbf{z}] &= \int \mathbf{g}(\mathbf{x})p(\mathbf{x}|\mathbf{z})d\mathbf{x} = \int \mathbf{g}(\mathbf{x})\frac{p(\mathbf{x}|\mathbf{z})}{q(\mathbf{x}|\mathbf{z})}q(\mathbf{x}|\mathbf{z})d\mathbf{x} \\ \mathbb{E}\left[\mathbf{g}(\mathbf{x})\frac{p(\mathbf{x}|\mathbf{z})}{q(\mathbf{x}|\mathbf{z})}\middle|\mathbf{z}\right] &\approx \sum_{i=1}^N \mathbf{g}(\mathbf{x}^{(i)})\tilde{w}^{(i)},\end{aligned}$$

where $\{\mathbf{x}^{(i)}\}_{i=1}^N \sim q(\mathbf{x}|\mathbf{z})$ and the *normalized importance weights* $\{\tilde{w}^{(i)}\}_{i=1}^N$ are generated by the following equations

$$\text{Unnormalized importance weights: } \{w^{(i)}\}_{i=1}^N = \frac{1}{N} \frac{p(\mathbf{x}^{(i)}|\mathbf{z})}{q(\mathbf{x}^{(i)}|\mathbf{z})}. \quad (4.8)$$

$$\text{Normalized importance weights: } \{\tilde{w}^{(i)}\}_{i=1}^N = \frac{w^{(i)}}{\sum_{j=1}^N w^{(j)}}. \quad (4.9)$$

This result holds for any function \mathbf{g} , and therefore an approximate distribution to the posterior distribution is given by

$$\mathbf{p}(\mathbf{x}|\mathbf{z}) \approx \sum_{i=1}^N \tilde{w}^{(i)} \delta_{\mathbf{x}^{(i)}}(x),$$

where $\delta_{\mathbf{x}^{(i)}} = \delta(\mathbf{x} - \mathbf{x}^{(i)})$ denotes the Dirac delta function centered in $\mathbf{x}^{(i)}$.

In contrast to importance sampling, *Sequential Importance Sampling* (SIS) is a SMC method that allows evaluation of the importance weights, defined by Equations (4.8)-(4.9), recursively in time. This makes the method suitable for estimation problems related to state-space models. A general SIS particle filter is presented in Algorithm 3 and is further explained here. Provided that for each subsequent time step the samples are drawn from the importance distribution $q(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})$, the weights are given by the following recursive updating formula

$$\text{Unnormalized weights: } w_k^{(i)} = \tilde{w}_{k-1}^{(i)} \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{q(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})}. \quad (4.10)$$

$$\text{Normalized weights: } \tilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^N w_k^{(j)}}. \quad (4.11)$$

Hence, the posterior state $p(\mathbf{x}_k|\mathbf{z}_{1:k})$, defined in equation (4.5), is approximated recursively according to

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \hat{p}(\mathbf{x}_k|\mathbf{z}_{1:k}) = \sum_{i=1}^N \tilde{w}_k^{(i)} \delta_{\mathbf{x}_k^{(i)}}(x_k).$$

Moreover, the unnormalized weights can be used for approximation of the marginal distribution $p(\mathbf{z}_{1:T})$ (or the likelihood $p(\mathbf{z}_{1:T}|\theta)$) according to

$$p(\mathbf{z}_{1:T}) = \prod_{k=1}^T p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) \approx \prod_{k=1}^T \left(\sum_{i=1}^N w_k^{(i)} \right).$$

If the transition model is chosen as the importance distribution, that is $p(\mathbf{x}_k|\mathbf{x}_{k-1}) = q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})$, then the updating formula for the weights simplifies to

$$\text{Unnormalized weights: } w_k^{(i)} = \tilde{w}_{k-1}^{(i)} p(\mathbf{z}_k|\mathbf{x}_k^{(i)}). \quad (4.12)$$

$$\text{Normalized weights: } \tilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^N w_k^{(j)}}. \quad (4.13)$$

Algorithm 3 Sequential Importance Sampling (SIS)

Input: $\mathbf{z}_{1:T}, N$

Output: $\{\mathbf{x}_{0:T}^{(i)}\}_{i=1}^N, \{\tilde{w}_{0:T}^{(i)}\}_{i=1}^N$

1. **Initialization**, $k=0$

- (a) Draw N particles from the the initial prior distribution.

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \quad i = 1, \dots, N$$

- (b) Set weights uniformly.

$$\tilde{w}_0^{(i)} = \frac{1}{N}, \quad i = 1, \dots, N$$

2. **Repeat for** $k = 1 : T$

- (a) Propagate particles according to the importance distribution.

$$\mathbf{x}_k^{(i)} \sim q(\mathbf{x}_k|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_{1:k}), \quad i = 1, \dots, N$$

- (b) Compute the weights according to equation (4.10).

$$w_k^{(i)} = \tilde{w}_{k-1}^{(i)} \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{q(\mathbf{x}_k^{(i)}|\mathbf{x}_{0:k-1}^{(i)}, \mathbf{z}_{1:k})}, \quad i = 1, \dots, N$$

- (c) Compute the normalized weights.

$$\tilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^N w_k^{(j)}}, \quad i = 1, \dots, N$$

SIS is an attractive method for recursive estimation problems, but it encounters some problems as the discrete time k increases [23]. As k increases, the distribution of the importance weights $w_k^{(i)}$

becomes more and more skewed. It means that most of the weights approach zero, leaving only a few particles to have non-zero importance weights. Thus, the algorithm fails to represent the posterior distributions of interest adequately. This problem is referred to as the *degeneracy problem*. To solve this problem, an additional step called resampling is normally introduced. When resampling is used, the algorithm previously described is instead called *Sequential Importance Resampling* (SIR). The SIR algorithm is presented in Algorithm 8 (see Appendix A.1). Resampling aims to eliminate particles with small weights by replacing them with replicated values of particles with larger importance weights [25]. There are several different resampling methods, such as *Multinomial resampling* and *Systematic resampling* [26]. A particular case of the SIR algorithm, is called the *Bootstrap filter*, presented in Algorithm 9 (see Appendix A.2). In the Bootstrap filter the transition model is chosen as the importance distribution, resulting in the simplified weight updating formula, described in (4.12)-(4.13).

Although resampling reduces the effects of the degeneracy problem, it introduces other practical problems [27]. As previously mentioned, particles with high importance weights are statistically selected multiple times. Consequently, the resultant sample contains many replicated values, which means that there is a loss of diversity among the particles. This problem is known as *sample impoverishment*. Moreover, since the diversity among the particles decreases with the resampling step, estimates that are based on the paths of the particles degenerate. Methods exist that counteract this problem. However, they are not considered in this thesis.

Models for Reflectivity

In this chapter, statistical models for describing the complex reflectivity, defined by (2.2), are presented. Section 5.1 presents a classical approach to the modeling of sea clutter: the K-distribution model. Moreover, the more general random walk model is discussed in Section 5.2 before Field's model is introduced in Section 5.3. Throughout the subsequent chapters, Field's model is employed for describing the sea clutter.

5.1 K-Distribution Model

Many radar systems focus primarily on the amplitude variations of the received signal and do not consider the specific phase information in their signal processing [3]. However, it crucial to recognize that in the context of coherent radar processing, a comprehensive understanding of both the amplitude of the reflected intensity and the associated phase information is essential. Coherent radar processing offers significant advantages, allowing for more precise target detection, tracking, and estimation of target properties, making it particularly valuable in a wide range of radar applications.

The Gaussian distribution is a reasonable model for describing amplitude statistics of sea clutter in low-resolution maritime radar systems [3]. However, when dealing with high-resolution radars and lower grazing angles, the Gaussian model begins to exhibit limitations. This is due to the fact that for higher resolution and lower grazing angles, the clutter is subject to much greater fluctuations (sea spikes) than are seen in Gaussian clutter.

An adequate non-Gaussian model should take into account that the sea waves consist of long gravitational waves and short capillary and wind waves [3]. In this context, the K-distribution provides a good model for the amplitude statistics of sea clutter [28]. The K-distribution is a compound distribution, which models the clutter amplitude E in terms of a product of a Rayleigh distribution and a Gamma distribution. Thus, the distribution $p(E)$ of the clutter amplitude of the non-Gaussian signal can be expressed as

$$p(E) = \int p(E|x)p(x)dx, \quad (5.1)$$

where x is the local mean power. The conditional distribution of E given x is provided by the following Rayleigh distribution

$$p(E|x) = \frac{2E}{x} \exp\left(-\frac{E^2}{x}\right), \quad 0 \leq E \leq \infty. \quad (5.2)$$

The distribution $p(x)$ is given by the Gamma distribution according to

$$p(x) = \frac{b^v}{\Gamma(v)} x^{v-1} \exp(-bx), \quad (5.3)$$

where, $\Gamma(\cdot)$ is the Gamma function. This distribution is characterised by a scale parameter, b , and a shape parameter, v , which depend on sea conditions and the radar parameters [3]. By substituting

the Rayleigh distribution (5.2) and the Gamma distribution (5.3) into Equation (5.1), the probability density of the clutter amplitude is given by the compound K-distribution according to

$$p(E) = \frac{4b^{(v+1)/2}E^v}{\Gamma(v)} K_{v-1} \left(2E\sqrt{b} \right),$$

where $K_{v-1}(\cdot)$ is the second-order modified Bessel function. The Rayleigh distributed component captures the fluctuations of the scatterers (short time scale) and the Gamma distributed component should reflect the intensity of the scattering (long time scale), in response to the underlying sea swell (gravity waves). It has been shown experimentally that K-distributions are able to accurately model sea clutter (compare [4] and references therein). However, it is imperative to note that the K-distribution model has its limitations. Specifically, it does not encompass the time evolution of the sea surface.

5.2 Random Walk Model

The Random Walk model for scattering is static in the sense that it does not model the time dependency of the sea clutter [1]. The derivation of the model starts from the central idea that the complex reflectivity, Ψ_t , is a sum of contributions over a population of independent scatterers. Thus, $\forall t \geq 0$

$$\Psi_t = \sum_{n=1}^{N_t} a_t^{(n)} e^{i\phi_t^{(n)}},$$

where $a_t^{(n)}$ is the amplitude and $\phi_t^{(n)}$ is the phase of the n -th scatterer. The phase $\phi_t^{(n)}$ is uniformly distributed over the interval $[0, 2\pi]$. The Random Walk model assumes that for fixed t , the amplitudes $a_t^{(n)}$ are independent and identically distributed. Moreover, the phases $\phi_t^{(n)}$ are independent for different n and all phases and amplitudes are independent. Additionally, the number of scatterers N_t is itself a random variable. Moreover, all processes are stationary, meaning the statistical properties do not change over time. Let the average number of scatterers be denoted as $\bar{N} = \mathbb{E}[N_t]$. If the amplitudes are normalized by $\bar{N}^{1/2}$ and $\bar{N} \rightarrow \infty$, the complex reflectivity becomes

$$\Psi_t = \lim_{\bar{N} \rightarrow +\infty} \sum_{n=1}^{N_t} \frac{a_t^{(n)}}{\bar{N}^{1/2}} e^{i\phi_t^{(n)}} = \lim_{\bar{N} \rightarrow +\infty} \left(\frac{N_t}{\bar{N}} \right)^{1/2} \lim_{\bar{N} \rightarrow +\infty} \sum_{n=1}^{N_t} \frac{a_t^{(n)}}{N_t^{1/2}} e^{i\phi_t^{(n)}} = x_t^{1/2} \gamma_t. \quad (5.4)$$

Here N_t is the actual number of scatterers at time t . In Equation (5.4) the reflectivity Ψ_t is factorized in two factors. This particular factorization is employed in Field's model, where the stochastic processes x_t (radar cross section) and γ_t (speckle) are solutions to stochastic differential equations (See Section 5.3). In the context of the random walk model, these processes are defined as follows

$$\begin{aligned} \text{Radar Cross Section: } x_t^{1/2} &= \lim_{\bar{N} \rightarrow +\infty} \left(\frac{N_t}{\bar{N}} \right)^{1/2}, \\ \text{Speckle: } \gamma_t &= \lim_{\bar{N} \rightarrow +\infty} \sum_{n=1}^{N_t} \frac{a_t^{(n)}}{N_t^{1/2}} e^{i\phi_t^{(n)}}. \end{aligned}$$

As previously discussed, the Random Walk model has a limitation in that it does not incorporate the dynamics of the sea clutter. Field's model, presented in the next section, accounts for the dynamic nature of the sea surface.

5.3 Field's Model

Field's model for reflectivity was first proposed in a series of works by Timothy Field *et al.* [4] and has been further clarified and evaluated by Clement Roussel *et al.* [1]. Field's model is a dynamical extension of the K-distribution model (see Section 5.1), but generalises from the Random Walk model (see Section 5.2) as well. The model expresses the sea clutter in terms of stochastic differential equations. The complex reflectivity Ψ_t can be factorized as

$$\Psi_t = x_t^{1/2} \gamma_t = x_t^{1/2} (\gamma_t^{(R)} + i\gamma_t^{(I)}),$$

where x_t and γ_t are solutions to the stochastic differential equations

$$\begin{cases} dx_t = \mathcal{A}(1 - x_t)dt + (2\frac{\mathcal{A}}{\alpha}x_t)^{\frac{1}{2}}dW_t^{(x)} \\ d\gamma_t^{(R)} = -\frac{1}{2}\mathcal{B}\gamma_t^{(R)}dt + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}dW_t^{(R)} \\ d\gamma_t^{(I)} = -\frac{1}{2}\mathcal{B}\gamma_t^{(I)}dt + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}dW_t^{(I)}. \end{cases} \quad (5.5)$$

Here, $W_t^{(x)}$, $W_t^{(R)}$ and $W_t^{(I)}$ are independent Brownian motions and \mathcal{A} , α and \mathcal{B} are constants. The process x_t is a Cox-Ingersoll-Ross (CIR) process, which describes the dynamics of the Radar Cross Section (RCS), and is parameterized by \mathcal{A} and α . Moreover, the speckle γ_t parameterized by \mathcal{B} is a complex Ornstein Uhlenbeck (OU) process, which describes the phase dynamics of the backscattered signal from the sea surface. Both \mathcal{A} and \mathcal{B} have the dimension of frequency (i.e., inversely proportional to correlation time) and for radar scattering $\mathcal{A} \ll \mathcal{B}$. Hence, the correlation time scale is much longer for the RCS than for the speckle. The constant α is a dimensionless shape parameter related to the sea state. The case $\mathcal{A} \equiv 0$ means a constant RCS which reduces to Rayleigh scattering. The parameter \mathcal{A} relates to the statistics of the scattering properties of the sea surface and is thus independent of the illuminating electromagnetic wave. The constant \mathcal{B} depends on the carrier wave.

Moreover, according to Field's theory [4] the stationary processes x_t , $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$ are gamma distributed and centered Gaussian distributed, respectively. So for all $t \geq 0$, they have the following probability density functions

$$\begin{cases} p(x_t = x) = \frac{\alpha^\alpha x^{\alpha-1} e^{-\alpha x}}{\Gamma(\alpha)} \\ p(\gamma^{(R)} = x) = p(\gamma^{(I)} = x) = \frac{1}{\sqrt{\pi}} e^{-x^2} \end{cases}$$

As stated in the introduction chapter, the aim of the thesis is to estimate the parameters \mathcal{A} , α and \mathcal{B} . The statistics of the sea clutter depend on the radar's viewing geometry together with prevailing conditions [10]. However, the viewing geometry is not taken into consideration. The goal is to estimate the parameters \mathcal{A} , α and \mathcal{B} only by means of the observed reflectivity Ψ_t .

6

Methodology

This Chapter explains the data and methods used for estimating the sea clutter parameters \mathcal{A} , α and \mathcal{B} in Field's model. First, Section 6.1 describes how to generate the data used for simulations and experiments. Based on this data, estimators for the stochastic processes x_t and γ_t are introduced in Section 6.2. These estimators are derived by Roussel in [1]. Thereafter, Section 6.3 presents two Metropolis-Hasting algorithms that uses the estimators as input. Finally, Section 6.4 presents a MH algorithm that incorporates a particle filter to enable estimation of the parameters.

6.1 Synthetic Time Series Data Ψ_t

The data Ψ_t , serving as input in Algorithm 7 and used for computing the estimators in Section 6.2, is synthetically generated. Indeed, it is produced by the means of Field's model described in Section 5.3. While an analytical solution is impossible to find for the CIR process x_t in Field's model, analytical solutions for the OU-processes $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$ can be obtained without the need for additional computational resources. Nevertheless, for consistency, it is chosen to approximate all solutions with numerical methods. Let $\Delta t = T/n$ for some positive integer n and final time T , and $t_k = k\Delta t$. Moreover, let Ψ_k , $\gamma_k^{(R)}$, $\gamma_k^{(I)}$ and x_k denote the numerical approximations to Ψ_{t_k} , $\gamma_{t_k}^{(R)}$, $\gamma_{t_k}^{(I)}$ and x_{t_k} , respectively. Hence, the synthetic complex reflectivity can be factorized according to

$$\Psi_k = x_k^{1/2}(\gamma_k^{(R)} + i\gamma_k^{(I)}), \quad k = 0, 1, 2, \dots, n.$$

In order to produce a numerical time series $\{\Psi_k\}_{k=0}^n$, one first needs to introduce appropriate initial values x_0 , $\gamma_0^{(R)}$ and $\gamma_0^{(I)}$. In addition, also the time step, Δt , the final time, T , and the parameters \mathcal{A} , α and \mathcal{B} must be specified.

The Euler-Maruyama scheme, given by Equation (3.2), is used to discretize the two SDEs describing $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$. This results in the following set of equations

$$\gamma_k^{(R)} = \gamma_{k-1}^{(R)} - \frac{1}{2}\mathcal{B}\gamma_{k-1}^{(R)}\Delta t + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}\Delta W_k^{(R)}, \quad k = 1, 2, \dots, n, \quad (6.1)$$

$$\gamma_k^{(I)} = \gamma_{k-1}^{(I)} - \frac{1}{2}\mathcal{B}\gamma_{k-1}^{(I)}\Delta t + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}\Delta W_k^{(I)}, \quad k = 1, 2, \dots, n. \quad (6.2)$$

Indeed, the Euler-Maruyama and Milstein schemes provide the same discretization for an OU process, which is why Euler-Maruyama is applied. In contrast to the speckle, $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$, the RCS x_t cannot be negative. Using the Euler-Maruyama scheme to discretize the RCS may result in negative values, which is clearly undesirably. To avoid this, one can use the Milstein scheme, given by Equation (3.3), that reduces the probability to obtain negative values. This results in the following discretization

$$x_k = x_{k-1} + \mathcal{A}(1 - x_{k-1})\Delta t + \left(\frac{2\mathcal{A}}{\alpha}x_{k-1}\right)^{\frac{1}{2}}\Delta W_k^{(x)}, \quad k = 1, 2, \dots, n.$$

The Initial values, x_0 , $\gamma_0^{(R)}$ and $\gamma_0^{(I)}$, and the time step, Δt , are kept constant throughout all simulations. They are compiled in Table 6.1.

Table 6.1: *Constants for Synthetic Data.*

x_0	$\gamma_0^{(R)}$	$\gamma_0^{(I)}$	Δt [s]
4	2	2	10^{-4}

6.2 Estimators for x_t and γ_t based on Ψ_t

This section introduces estimators for x_t and γ_t , which can be used as inputs in the Metropolis-Hastings methods, presented in Algorithm 5 and 4, respectively. These algorithms are designed for estimating the sea clutter parameters \mathcal{A} , α and \mathcal{B} . In practice, it is only possible to observe the complex reflectivity Ψ_t [1], that is, the RCS x_t and speckle $\gamma_t^{(R)}$, $\gamma_t^{(I)}$ are unobservable, hidden processes. Hence, these processes need to be estimated. It is possible to estimate x_t and γ_t based on the increments of a discrete time series of the reflectivity $\{\Psi_k, k = 1, 2, \dots, n\}$. Roussel [1] proposes three estimators for x_t . Naturally, an estimator for x_t results in an estimator for γ_t since $\gamma_t = \Psi_t/x_t^{1/2}$.

The derivations of the three estimators \bar{x}_t , \hat{x}_t and \tilde{x}_t are found in Appendix B.1, B.2 and B.3, respectively. Both \bar{x}_t and \tilde{x}_t are based on the increments $\Delta_{t_k}\Psi = \Psi_{t_k} - \Psi_{t_{k-1}}$ of the complex reflectivity. The estimators \hat{x}_t and \bar{x}_t are almost identical with the exception that \tilde{x}_t uses an estimator $\tilde{\mathcal{B}}_\Psi$ for \mathcal{B} . Consequently, only \tilde{x}_t is suitable as input to Algorithm 5, designed for estimating \mathcal{A} and α with MH. This distinction arises since the parameter \mathcal{B} is unknown while the estimator $\tilde{\mathcal{B}}_\Psi$ is computable. Also, the estimator \hat{x}_t , which is based on the intensity $z_t = |\Psi_t|^2$, can be used for Bayesian estimation. To summarize, only \hat{x}_t and \tilde{x}_t are possible to obtain from real data. However, \bar{x}_t can be computed since synthetic data is used and consequently the real parameter \mathcal{B} , used to produce the data, is known. The estimators are

$$\bar{x}_t = \frac{1}{\mathcal{B}\Delta t N} \sum_{t_k \in \Delta_t} |\Delta_{t_k}\Psi|^2, \quad (6.3)$$

$$\hat{x}_t = \frac{1}{N} \sum_{t_k \in \Delta_t} z_{t_k}, \quad (6.4)$$

$$\tilde{x}_t = \frac{1}{\tilde{\mathcal{B}}_\Psi \Delta t N} \sum_{t_k \in \Delta_t} |\Delta_{t_k}\Psi|^2. \quad (6.5)$$

In this context, Δ_t represents a time window centered at time t , N denotes the number of elements within that window, $|\Delta_{t_k}\Psi|$ is the reflectivity increments within the time window and Δt is the simulation time step. Moreover, $\tilde{\mathcal{B}}_\Psi$ is the estimator for \mathcal{B} , given by the following equation

$$\tilde{\mathcal{B}}_\Psi = \frac{1}{m\Delta t N} \sum_{i=1}^m \sum_{t_k \in \Delta_i} |\Delta_{t_k}\Psi|^2.$$

Here, Δ_i is a time window centered at time t_i and m is the number of time windows.

To compare and evaluate the estimators, Roussel estimates the mean of 10 sampled trajectories of Ψ_t . In this thesis, instead 20 trajectories are sampled according to the discretizations in Section 6.1. The sea clutter parameters are set to $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. Furthermore, the final time is set to $T = 300$ seconds. According to Roussel, such a large T is required to ensure that $\tilde{\mathcal{B}}_\Psi$ is close to \mathcal{B} . By computing the average of $\tilde{\mathcal{B}}_\Psi$, generated by 20 trajectories, for multiple durations T , it can be verified in Table 6.2 that $T = 300$ seconds should be sufficient. However, it is worth noting

that such a large T make the estimators impractical in real-world applications due to the extended computational demands. The estimators are compared in Section 7.1.

Table 6.2: *Duration time T and corresponding average \bar{B}_Ψ for 20 trajectories.*

T [s]	10	50	100	150	200	250	300
\bar{B}_Ψ [Hz]	121.02	107.82	105.60	102.06	97.46	101.30	101.03

6.3 Bayesian Estimation of \mathcal{A} , α , \mathcal{B} using Metropolis-Hastings

The aim of this section is to apply the Bayesian approach, proposed by Roussel in his work [1], to estimate the clutter parameters in Field's model. This includes using the best estimator for x_t and γ_t , that is based on the complex reflectivity Ψ_t , derived in Section 6.2. The Bayesian estimation is performed with Metropolis-Hastings. First, the methodology for estimating \mathcal{B} is described in Section 6.3.1. Then, a similar method for approximating \mathcal{A} and α is proposed in Section 6.3.2.

6.3.1 Estimation of \mathcal{B}

This section describes a Metropolis-Hastings method, presented in Algorithm 4, for estimating \mathcal{B} given an estimator for γ_t . Consider for simplicity the real part of the speckle process in Field's model (5.5), which includes the clutter parameter \mathcal{B} . Recall that the equation is defined as follows

$$d\gamma_t^{(R)} = -\frac{1}{2}\mathcal{B}\gamma_t^{(R)}dt + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}dW_t^{(R)}.$$

According to Bayes formula, the posterior distribution of \mathcal{B} is proportional to the likelihood times the prior

$$p(\mathcal{B}|\gamma^{(R)}) \propto \mathcal{L}(\mathcal{B}|\gamma^{(R)})\pi(\mathcal{B}).$$

Here, $\gamma^{(R)}$ denotes a time series of the best-performing estimator, introduced in Section 6.2 (the performance of the estimators are presented in Section 7.1). Based on the knowledge of the range of possible values for \mathcal{B} , Roussel proposes a uniform prior over $[10, 1000]$. The same prior is applied here. Thus, the prior has the distribution $\pi(\mathcal{B}) \sim \mathcal{U}(10, 1000)$.

Given an estimated time series $\gamma^{(R)} = \{\gamma_{t_0}^{(R)}, \gamma_{t_1}^{(R)}, \dots, \gamma_{t_n}^{(R)}\}$, the likelihood can be expressed according to

$$\mathcal{L}(\mathcal{B}|\gamma^{(R)}) = p_B(\gamma_{t_0}^{(R)}) \prod_{k=1}^n p_B(\gamma_{t_k}^{(R)}|\gamma_{t_{k-1}}^{(R)}),$$

where $p_B(\gamma_{t_k}^{(R)}|\gamma_{t_{k-1}}^{(R)})$ denotes the transition probabilities. To determine the transition probabilities, recall that the Euler-Maruyama discretization of the SDE above is given by

$$\gamma_{t_k}^{(R)} = \gamma_{t_{k-1}}^{(R)} - \frac{1}{2}\mathcal{B}\gamma_{t_{k-1}}^{(R)}\Delta t + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}\Delta W_{t_k}^{(R)}, \quad k = 1, 2, \dots, N.$$

By using this Euler-Maruyama discretization, the following transition probabilities are obtained

$$p_B(\gamma_{t_k}^{(R)}|\gamma_{t_{k-1}}^{(R)}) = \frac{1}{\sqrt{\pi\mathcal{B}\Delta t}} \exp\left(-\frac{\left(\gamma_{t_k}^{(R)} - \gamma_{t_{k-1}}^{(R)}(1 - \mathcal{B}\Delta t/2)\right)^2}{\mathcal{B}\Delta t}\right).$$

Hence, the likelihood becomes

$$\mathcal{L}(\mathcal{B}|\gamma^{(R)}) = \frac{1}{\sqrt{\pi}} e^{-\gamma_{t_0}^{(R)2}} \prod_{k=1}^n \frac{1}{\sqrt{\pi\mathcal{B}\Delta t}} \exp\left(-\frac{\left(\gamma_{t_k}^{(R)} - \gamma_{t_{k-1}}^{(R)}(1 - \mathcal{B}\Delta t/2)\right)^2}{\mathcal{B}\Delta t}\right).$$

This yields the log-likelihood

$$\ell(\mathcal{B}|\gamma^{(R)}) = -\frac{1}{2} (\log(\pi) + n \log(\pi B \Delta t)) - \gamma_{t_0}^{(R)2} - \frac{1}{B \Delta t} \sum_{k=1}^n \left(\gamma_{t_k}^{(R)} - \gamma_{t_{k-1}}^{(R)} (1 - B \Delta t / 2) \right)^2. \quad (6.6)$$

As a last step, one needs to choose an appropriate proposal distribution from which the candidates \mathcal{B}^* are drawn. This distribution is chosen as an adaptive normal distribution, resulting in a symmetric distribution. Hence, the acceptance probability depends only on the likelihood. More specifically, the proposal distribution is chosen as

$$\begin{cases} q_{\mathcal{B}}(\mathcal{B}^*|\mathcal{B}_{j-1}) \sim \mathcal{N}(\mu_{\mathcal{B}}, \sigma_{\mathcal{B}}), \\ \mu_{\mathcal{B}} = \mathcal{B}_{j-1}, \\ \sigma_{\mathcal{B}} = 0.5 + 7.5j^{-1/2}, \end{cases} \quad (6.7)$$

where $j = 1, \dots, M$ is the current iteration in the MH algorithm. The variance σ^2 decreases for every iteration, resulting in that more candidates \mathcal{B}^* are accepted.

Algorithm 4 Metropolis-Hastings for Estimating \mathcal{B}

Input: $\gamma^{(R)}, \mathcal{B}_0, M$

Output: $\mathcal{B}_{1:M}$

Repeat for $j = 1 : M$

1. **Proposal**

- (a) Sample a candidate \mathcal{B}^* from the proposal distribution $q_{\mathcal{B}}(\mathcal{B}^*|\mathcal{B}_{j-1})$ according to equation (6.7).
- (b) Calculate the log-likelihoods: $\ell(\mathcal{B}^*|\gamma^{(R)})$ and $\ell(\mathcal{B}_{j-1}|\gamma^{(R)})$ according to equation (6.6).
- (c) Compute the acceptance probability $\log \alpha = \min \{0, \ell(\mathcal{B}^*|\gamma^{(R)}) - \ell(\mathcal{B}_{j-1}|\gamma^{(R)})\}$.

2. **Acceptance or Rejection**

- (a) Draw $u \sim \mathcal{U}(0, 1)$.
 - (b) If $u \leq e^{\log \alpha}$, accept \mathcal{B}^* and set $\mathcal{B}_j = \mathcal{B}^*$.
 - (c) If $u > e^{\log \alpha}$, reject \mathcal{B}^* and set $\mathcal{B}_j = \mathcal{B}_{j-1}$.
-

6.3.2 Estimation of A and α

In the same way as for \mathcal{B} , also \mathcal{A} and α can be estimated by the MH method. The proposed algorithm is compiled in Algorithm 5. Consider the SDE, describing the RCS, in Field's model (5.5)

$$dx_t = A(1 - x_t)dt + \left(2\frac{A}{\alpha}x_t\right)^{\frac{1}{2}} dW_t^{(x)}.$$

Bayes formula (4.1) is applied to estimate the clutter parameters, \mathcal{A} and α , given an estimated time series x of x_t . According to Bayes formula, the posterior distribution of \mathcal{A} and α is proportional to the likelihood times the prior

$$p(\mathcal{A}, \alpha|x) \propto \mathcal{L}(\mathcal{A}, \alpha|x)\pi(\mathcal{A})\pi(\alpha).$$

In contrast to Roussel [1], who suggests a uniform prior for \mathcal{A} and a Gamma prior for α , this thesis adopts a uniform prior for both parameters. It is not evident why Roussel suggests a Gamma prior for α . Here, it is presumed that there is no prior knowledge of the parameters except the range of possible values for \mathcal{A} and α . Hence, the priors are assumed to be distributed according to $\pi(\mathcal{A}) \sim \mathcal{U}(0.1, 10)$ and $\pi(\alpha) \sim \mathcal{U}(0, 15)$.

Let $x = \{x_{t_0}, x_{t_1}, \dots, x_{t_n}\}$ be an estimated time series, provided by Equation (6.5) or (6.4). Then, the likelihood can be expressed as

$$\mathcal{L}(\mathcal{A}, \alpha | x) = p_{\mathcal{A}, \alpha}(x_{t_0}) \prod_{k=1}^n p_{\mathcal{A}, \alpha}(x_{t_k} | x_{t_{k-1}}),$$

where $p_{\mathcal{A}, \alpha}(x_{t_k} | x_{t_{k-1}})$ denotes the transition probabilities. By using Euler-Maruyama discretization for the SDE, the transition probabilities become

$$p_{\mathcal{A}, \alpha}(x_{t_k} | x_{t_{k-1}}) = \frac{\sqrt{\alpha}}{\sqrt{4\pi x_{t_{k-1}} \mathcal{A} \Delta t}} \exp\left(-\frac{\alpha (x_{t_k} - \mathcal{A} \Delta t - (1 - \mathcal{A} \Delta t)x_{t_{k-1}})^2}{4\mathcal{A} \Delta t x_{t_{k-1}}}\right).$$

Hence, the likelihood can be written as

$$\mathcal{L}(\mathcal{A}, \alpha | x) = \frac{\alpha^\alpha x_{t_0}^{\alpha-1} e^{-\alpha x_{t_0}}}{\Gamma(\alpha)} \prod_{k=1}^n \frac{\sqrt{\alpha}}{\sqrt{4\pi x_{t_{k-1}} \mathcal{A} \Delta t}} \exp\left(-\frac{\alpha (x_{t_k} - \mathcal{A} \Delta t - (1 - \mathcal{A} \Delta t)x_{t_{k-1}})^2}{4\mathcal{A} \Delta t x_{t_{k-1}}}\right).$$

Further, the log-likelihood becomes

$$\begin{aligned} \ell(\mathcal{A}, \alpha | x) &= \alpha(\log \alpha - x_{t_0}) + (\alpha - 1) \log x_{t_0} - \log \Gamma(\alpha) + \frac{n}{2} \log \frac{\alpha}{4\pi \mathcal{A} \Delta t} \\ &\quad + \sum_{k=1}^n \left[\frac{1}{2} \log x_{t_{k-1}} + \frac{\alpha (x_{t_k} - \mathcal{A} \Delta t - (1 - \mathcal{A} \Delta t)x_{t_{k-1}})^2}{4\mathcal{A} \Delta t x_{t_{k-1}}} \right]. \end{aligned} \quad (6.8)$$

Again, one needs to choose appropriate proposal distributions from which the candidates \mathcal{A}^* and α^* are drawn. Similar to \mathcal{B} , these distributions are chosen as adaptive normal distributions, resulting in symmetric distributions. Thus, the acceptance probability in the MH algorithm depends only on the likelihood. Particularly, the proposals are chosen as

$$\begin{cases} q_{\mathcal{A}}(\mathcal{A}^* | \mathcal{A}_{j-1}) \sim \mathcal{N}(\mu_{\mathcal{A}}, \sigma_{\mathcal{A}}) \\ \mu_{\mathcal{A}} = \mathcal{A}_{j-1} \\ \sigma_{\mathcal{A}} = 0.0001 + 1.5 * j^{-(3/5)} \end{cases} \quad (6.9)$$

$$\begin{cases} q_{\alpha}(\alpha^* | \alpha_{j-1}) \sim \mathcal{N}(\mu_{\alpha}, \sigma_{\alpha}) \\ \mu_{\alpha} = \alpha_{j-1} \\ \sigma_{\alpha} = 0.0001 + 1.5 * j^{-(3/5)} \end{cases} \quad (6.10)$$

where $j = 1, \dots, M$ is the current iteration in the MH algorithm.

Algorithm 5 Metropolis-Hastings for Estimating \mathcal{A} and α

Input: $x, \mathcal{A}_0, \alpha_0, M$ **Output:** $\mathcal{A}_{1:M}, \alpha_{1:M}$ **Repeat for** $j = 1 : M$ **1. Proposal**

- (a) Sample a candidate $\mathcal{A}^* \sim q_{\mathcal{A}}(\mathcal{A}^*|\mathcal{A}_{j-1})$ according to equation (6.9).
- (b) Sample a candidate $\alpha^* \sim q_{\alpha}(\alpha^*|\alpha_{j-1})$ according to equation (6.10).
- (c) Calculate the log-likelihoods: $\ell(\mathcal{A}^*, \alpha^*|x)$ and $\ell(\mathcal{A}_{j-1}, \alpha_{j-1}|x)$ according to equation (6.8).
- (d) Compute the acceptance probability $\log \alpha = \min \{0, \ell(\mathcal{A}^*, \alpha^*|x) - \ell(\mathcal{A}_{j-1}, \alpha_{j-1}|x)\}$.

2. Acceptance or Rejection

- (a) Draw $u \sim \mathcal{U}(0, 1)$.
 - (b) If $u \leq e^{\log \alpha}$, accept \mathcal{A}^*, α^* and set $(\mathcal{A}_j, \alpha_j) = (\mathcal{A}^*, \alpha^*)$
 - (c) If $u > e^{\log \alpha}$, reject \mathcal{A}^*, α^* and set $(\mathcal{A}_j, \alpha_j) = (\mathcal{A}_{j-1}, \alpha_{j-1})$.
-

6.4 Estimation of $\mathcal{A}, \alpha, \mathcal{B}$ using Bayesian Filtering and Metropolis-Hastings

In the previous section, a time series of 300s of the complex reflectivity was needed in order to obtain accurate estimators for γ_t and x_t , which then were used to compute the posterior distributions of \mathcal{B} and \mathcal{A}, α , respectively. Another approach, is to apply the observed complex reflectivity Ψ_t to a particle filter, allowing for computations based on shorter time series, possibly with a wider posterior.

Let $\Psi_{t_k} = x_{t_k}^{\frac{1}{2}} \gamma_{t_k} + v_k$ denote the observed complex reflectivity, where t_k are sampling times and v_k are i.i.d. complex Gaussian random variables with covariance σI for some $\sigma > 0$. The noise v_k accounts for uncertainty in the observations, which makes the particle filter more robust to imperfect data. Bayes Formula and independence of \mathcal{A}, α and \mathcal{B} imply

$$p(\mathcal{A}, \alpha, \mathcal{B}|\Psi) \propto p(\Psi|\mathcal{A}, \alpha, \mathcal{B})p(\mathcal{A}, \alpha, \mathcal{B}) = p(\Psi|\mathcal{A}, \alpha, \mathcal{B})p(\mathcal{A})p(\alpha)p(\mathcal{B}),$$

where $p(\Psi|\mathcal{A}, \alpha, \mathcal{B})$ is the likelihood and $p(\mathcal{A})$, $p(\alpha)$, and $p(\mathcal{B})$ are prior distributions. By the Markov property of Ψ , one can write:

$$p(\Psi|\mathcal{A}, \alpha, \mathcal{B}) = p(\Psi_0) \prod_{k=1}^n p(\Psi_k|\Psi_{k-1}, \mathcal{A}, \alpha, \mathcal{B}). \quad (6.11)$$

To avoid dealing with complicated integrals with Bessel functions appearing in the transition probabilities for the Cox-Ingersoll-Ross process x_t , the transition probabilities $p(\Psi_k|\Psi_{k-1}, \mathcal{A}, \alpha, \mathcal{B})$ can be approximated directly with a particle filter. Once the likelihood can be determined, the MH method can be implemented to find the posterior distribution.

In this section, a method for this approach is proposed. First, the state-space model is defined. Second, the particle filter is presented followed by a presentation of how to evaluate its performance by using kernel estimation. It includes an assessment of the preferred parameter values in the particle filter. Lastly, the complete algorithm for computing the posterior distribution $p(\mathcal{A}, \alpha, \mathcal{B}|\Psi)$ is presented in Section 6.4.3.

6.4.1 State-Space Model

Recall that the process x_t is a CIR process and $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$ are OU processes, which satisfy the Markov property. After discretization, the processes x_t , $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$ can all be described by state-space models. A state-space model is described by a prior distribution, transition model and a measurement model (see Section 4.3.1). Here, the complex reflectivity constitute the measurements. Unfortunately, there is no transition model that directly describes its evolution over time. Instead, the transition models for the three processes x_t , $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$ are combined in order to determine how the complex reflectivity is evolving over time. First, let $\{\mathbf{y}_k\}_{k=0}^n$ and $\{\Psi_k\}_{k=1}^n$ denote the hidden states and the measured states (for the complex reflectivity) up to the final time step n , respectively. The prior distributions for the three processes are chosen as being uniform. More specifically

$$p(x_0) = \mathcal{U}(1, 5), \quad (6.12)$$

$$p(\gamma_0^{(R)}) = \mathcal{U}(-1, 3), \quad (6.13)$$

$$p(\gamma_0^{(I)}) = \mathcal{U}(-1, 3). \quad (6.14)$$

The SDEs for x_t , $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$ are solved numerically in Section 6.1. These discretizations describe how the processes are evolving over time. Thus, these numerical solutions are considered the transition models

$$p(x_k | x_{k-1}) = x_{k-1} + \mathcal{A}(1 - x_{k-1})\Delta t + \left(\frac{2\mathcal{A}}{\alpha}x_{k-1}\right)^{\frac{1}{2}} \Delta W_k^{(x)}, \quad (6.15)$$

$$p(\gamma_k^{(R)} | \gamma_{k-1}^{(R)}) = \gamma_{k-1}^{(R)} - \frac{1}{2}\mathcal{B}\gamma_{k-1}^{(R)}\Delta t + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}\Delta W_k^{(R)}, \quad (6.16)$$

$$p(\gamma_k^{(I)} | \gamma_{k-1}^{(I)}) = \gamma_{k-1}^{(I)} - \frac{1}{2}\mathcal{B}\gamma_{k-1}^{(I)}\Delta t + \frac{1}{\sqrt{2}}\mathcal{B}^{\frac{1}{2}}\Delta W_k^{(I)}. \quad (6.17)$$

The measurement model that relates measurements Ψ_k to the states \mathbf{y}_k is chosen as a Gaussian distribution

$$p(\Psi_k | \mathbf{y}_k) = \mathcal{N}(\mathbf{y}_k, \sigma^2), \quad (6.18)$$

where, $\sigma^2 > 0$ is a complex variance. It is explained in Section 6.4.4 how to choose an appropriate value for this parameter.

The previously described equations constitute together the complex nonlinear state-space system that describes the process Ψ_t (complex reflectivity). As stated earlier, the distribution of interest is the likelihood $p(\Psi | \mathcal{A}, \alpha, \mathcal{B})$. The factors $p(\Psi_k | \Psi_{k-1}, \mathcal{A}, \alpha, \mathcal{B})$ in Equation (6.11) can be estimated with a particle filter, in which the state-space equations are highly important.

6.4.2 Particle Filter

The purpose of the particle filter is to estimate the likelihood $p(\Psi | \mathcal{A}, \alpha, \mathcal{B})$, and the filter is designed accordingly. The suggested algorithm for estimating the likelihood $p(\Psi | \mathcal{A}, \alpha, \mathcal{B})$ is presented in Algorithm 6 and further explained here. Again, let $\{\mathbf{y}_k\}_{k=0}^n$ and $\{\Psi_k\}_{k=1}^n$ denote the hidden states and the measured states of the complex reflectivity up to the final time step n , respectively. Hence, the theory in Section 4.3.2, regarding particle filters, implies that the likelihood can be approximated by

$$p(\Psi | \mathcal{A}, \alpha, \mathcal{B}) = \prod_{k=1}^n p(\Psi_k | \Psi_{k-1}) \approx \prod_{k=1}^n \left(\sum_{i=1}^N w_k^{(i)} \right), \quad (6.19)$$

where N is the number of particles and $w_k^{(i)}$ are the unnormalized importance weights given by

$$w_k^{(i)} = \tilde{w}_{k-1}^{(i)} \frac{p(\Psi_k | \mathbf{y}_k^{(i)})p(\mathbf{y}_k^{(i)} | \mathbf{y}_{k-1}^{(i)})}{q(\mathbf{y}_k^{(i)} | \mathbf{y}_{0:k-1}^{(i)}, \Psi_{1:k})}.$$

The importance distribution $q(\mathbf{y}_k^{(i)} | \mathbf{y}_{0:k-1}^{(i)}, \Psi_{1:k})$ is chosen as the transition model $p(\mathbf{y}_k^{(i)} | \mathbf{y}_{k-1}^{(i)})$ and hence, the weight formula reduces to

$$w_k^{(i)} = \tilde{w}_{k-1}^{(i)} p(\Psi_k | \mathbf{y}_k^{(i)}).$$

Here, $\tilde{w}_{k-1}^{(i)}$ are the normalized importance weights and $p(\Psi_k | \mathbf{y}_k^{(i)})$ denotes the measurement model described in the previous Section by (6.18). To clarify, the measurement model is the likelihood of observing the complex reflectivity Ψ_k given predictions $\mathbf{y}_k^{(i)}$.

The predictions are particles, provided by combining the transition models defined in Section 6.4.1. In particular, for each process x_t , $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$, N particles $x_k^{(i)}$, $\gamma_k^{(R)(i)}$ and $\gamma_k^{(I)(i)}$ are drawn from the prior distribution (6.12), (6.13) and (6.13), respectively. Then, these particles are propagated according to their transition models (6.15), (6.16) and (6.17). The predictions $\mathbf{y}_k^{(i)}$ are then obtained by

$$\mathbf{y}_k^{(i)} = (\mathbf{x}_k^{(i)})^{\frac{1}{2}} \left(\gamma_k^{(R)(i)} + i \gamma_k^{(I)(i)} \right). \quad (6.20)$$

It is chosen to not include a resampling step in this algorithm. As stated in Section 4.3.2, resampling aims to eliminate particles with small weights by replacing them with replicated values of particles with larger importance weights. This can cause diversity among the particles, which in turn can result in a biased likelihood approximation. In other words, when incorporating the particle filter in the Metropolis-Hastings algorithm some suggested parameter values \mathcal{A}^* , α^* and \mathcal{B}^* may seem to be better suggestions than they really are.

Algorithm 6 Particle Filter for Likelihood Approximation

Input: $\Psi_{1:n}$, N , σ , \mathcal{A} , α , \mathcal{B}

Output: $p(\Psi | \mathcal{A}, \alpha, \mathcal{B})$

1. **Initialization**, $k=0$

- (a) Draw N particles $\mathbf{x}_0^{(i)}$, $\gamma_0^{(R)(i)}$, $\gamma_0^{(I)(i)}$ from the the prior distributions (6.12), (6.13), (6.14), respectively.
- (b) Set initial weights uniformly

$$\{\tilde{w}_0^{(i)}\}_{i=1}^N = \frac{1}{N}.$$

2. **Repeat for** $k = 1 : n$

For $i = 1 : N$

- (a) Propagate the particles $\mathbf{x}_k^{(i)}$, $\gamma_k^{(R)(i)}$, $\gamma_k^{(I)(i)}$ according to (6.15), (6.16), (6.17), respectively.
- (b) Then, set $\mathbf{y}_k^{(i)}$ according to (6.20).
- (c) Compute $p(\Psi_k | \mathbf{y}_k^{(i)})$ according to the measurement model (6.18).
- (d) Compute the unnormalized importance weights $w_k^{(i)}$ according to

$$w_k^{(i)} = \tilde{w}_{k-1}^{(i)} p(\Psi_k | \mathbf{y}_k^{(i)}).$$

- (e) Set the normalized weights uniformly

$$\tilde{w}_k^{(i)} = \frac{1}{N}.$$

- 3. Compute the likelihood $p(\Psi | \mathcal{A}, \alpha, \mathcal{B})$ according to (6.19).
-

6.4.3 Particle Filter within Metropolis-Hastings

As discussed previously in Section 6.3, the posteriors $p(\mathcal{A}, \alpha|x)$ and $p(\mathcal{B}|\gamma^{(R)})$ can be computed using Metropolis-Hastings. However, this approach requires estimators for x and $\gamma^{(R)}$ (see Section 6.2). Instead, this section suggests an algorithm for computing the full posterior $p(\mathcal{A}, \alpha, \mathcal{B}|\Psi)$, which is based only on a time series of the observed reflectivity Ψ and not on estimators for x and $\gamma^{(R)}$. The proposed algorithm, presented in Algorithm 7, is an adaptive Metropolis-Hastings method that incorporates a particle filter.

The algorithm starts with the initializing step. As before, the priors are assumed to be $\pi(\mathcal{A}) \sim \mathcal{U}(0.1, 10)$, $\pi(\alpha) \sim \mathcal{U}(0, 15)$ and $\pi(\mathcal{B}) \sim \mathcal{U}(10, 1000)$. Initial values \mathcal{A}_0, α_0 and \mathcal{B}_0 are drawn from these distributions. In addition, starting values for the adaptive variances are set to $\sigma_{\mathcal{A}}^2 = 0.05^2$, $\sigma_{\alpha}^2 = 0.05^2$ and $\sigma_{\mathcal{B}}^2 = 20^2$.

As mentioned in Section 4.2 it can be challenging to find appropriate proposals that make the Markov chains converge fast. Therefore, in Algorithm 7, adaptive proposals are applied. These proposals are influenced by the adaptive proposal proposed in Section 4.2.1. Indeed, the proposal distributions, are chosen as normal distributions with adaptive variances, which makes it easier for the posterior distribution $p(\mathcal{A}, \alpha, \mathcal{B}|\Psi)$ to converge to the true posterior. The proposal distributions are given by

$$q_{\mathcal{A}}(\mathcal{A}^*|\mathcal{A}_{j-1}) \sim \mathcal{N}(\mathcal{A}_{j-1}, \sigma_{\mathcal{A}}^2), \quad (6.21)$$

$$q_{\alpha}(\alpha^*|\alpha_{j-1}) \sim \mathcal{N}(\alpha_{j-1}, \sigma_{\alpha}^2), \quad (6.22)$$

$$q_{\mathcal{B}}(\mathcal{B}^*|\mathcal{B}_{j-1}) \sim \mathcal{N}(\mathcal{B}_{j-1}, \sigma_{\mathcal{B}}^2). \quad (6.23)$$

$\sigma_{\mathcal{A}}^2, \sigma_{\alpha}^2$ and $\sigma_{\mathcal{B}}^2$ are kept constant for the first 100 iterations. Then, the adaptive process begins. At each iteration $j > 100$, the variance of the previous 100 values of the three Markov chains $\mathcal{A}_{j-101:j-1}$, $\alpha_{j-101:j-1}$ and $\mathcal{B}_{j-101:j-1}$ are computed, respectively. If the new variance is smaller than the current variance, the variance is updated according to

$$\sigma_{\mathcal{A}}^2 = \text{Var}(\mathcal{A}_{j-101:j-1}), \quad \text{if } \text{Var}(\mathcal{A}_{j-101:j-1}) < \sigma_{\mathcal{A}}^2, \quad (6.24)$$

$$\sigma_{\alpha}^2 = \text{Var}(\alpha_{j-101:j-1}), \quad \text{if } \text{Var}(\alpha_{j-101:j-1}) < \sigma_{\alpha}^2, \quad (6.25)$$

$$\sigma_{\mathcal{B}}^2 = \text{Var}(\mathcal{B}_{j-101:j-1}), \quad \text{if } \text{Var}(\mathcal{B}_{j-101:j-1}) < \sigma_{\mathcal{B}}^2. \quad (6.26)$$

Otherwise, the variance is unchanged. Candidates \mathcal{A}^*, α^* and \mathcal{B}^* are then sampled from their corresponding distributions provided by Equations (6.21)-(6.26).

Since the prior distributions are uniform and the proposal distributions are adaptive normal, which results in symmetric distributions, the acceptance probability only depends on the log-likelihoods according to

$$\log \alpha = \min \{0, \ell(\mathcal{A}^*, \alpha^*, \mathcal{B}^*|\Psi_{1:n}) - \ell(\mathcal{A}_{j-1}, \alpha_{j-1}, \mathcal{B}_{j-1}|\Psi_{1:n})\}$$

These likelihoods are approximated by the particle filter presented in Algorithm 6 .

Algorithm 7 Particle Filter within Metropolis-Hastings for Estimating \mathcal{A} , α and \mathcal{B}

Input: $\Psi_{1:n}, M$ **Output:** $(\mathcal{A}_j, \alpha_j, \mathcal{B}_j)_{j=1}^M$,**1. Initialization**

- (a) Draw $\mathcal{A}_0 \sim \pi(\mathcal{A})$ and set $\sigma_{\mathcal{A}} = 0.05$
- (b) Draw $\alpha_0 \sim \pi(\alpha)$ and set $\sigma_{\alpha} = 0.05$
- (c) Draw $\mathcal{B}_0 \sim \pi(\mathcal{B})$ and set $\sigma_{\mathcal{B}} = 20$

2. Repeat for $j = 1 : M$ **1. Adaptive Variance****(a) Variance for \mathcal{A}**

if $j > 100$ & $\text{Var}(\mathcal{A}_{j-101:j-1}) < \sigma_{\mathcal{A}}^2$ then
set $\sigma_{\mathcal{A}}^2 = \text{Var}(\mathcal{A}_{j-101:j-1})$
else
set $\sigma_{\mathcal{A}}^2 = \sigma_{\mathcal{A}}^2$

(b) Variance for α

if $j > 100$ & $\text{Var}(\alpha_{j-101:j-1}) < \sigma_{\alpha}^2$ then
set $\sigma_{\alpha}^2 = \text{Var}(\alpha_{j-101:j-1})$
else
set $\sigma_{\alpha}^2 = \sigma_{\alpha}^2$

(c) Variance for \mathcal{B}

if $j > 100$ & $\text{Var}(\mathcal{B}_{j-101:j-1}) < \sigma_{\mathcal{B}}^2$ then
set $\sigma_{\mathcal{B}}^2 = \text{Var}(\mathcal{B}_{j-101:j-1})$
else
set $\sigma_{\mathcal{B}}^2 = \sigma_{\mathcal{B}}^2$

3. Proposals

- (a) Draw \mathcal{A}^* from $\mathcal{N}(\mathcal{A}_{j-1}, \sigma_{\mathcal{A}}^2)$.
- (b) Draw α^* from $\mathcal{N}(\alpha_{j-1}, \sigma_{\alpha}^2)$.
- (c) Draw \mathcal{B}^* from $\mathcal{N}(\mathcal{B}_{j-1}, \sigma_{\mathcal{B}}^2)$.

3. Acceptance Probability

- (a) Compute the log-likelihoods $\ell(\mathcal{A}^*, \alpha^*, \mathcal{B}^* | \Psi_{1:n})$ and $\ell(\mathcal{A}_{j-1}, \alpha_{j-1}, \mathcal{B}_{j-1} | \Psi_{1:n})$ according to algorithm 6.
- (b) Calculate the acceptance probability: $\log \alpha = \min \{0, \ell(\mathcal{A}^*, \alpha^*, \mathcal{B}^* | \Psi_{1:n}) - \ell(\mathcal{A}_{j-1}, \alpha_{j-1}, \mathcal{B}_{j-1} | \Psi_{1:n})\}$.

4. Acceptance or Rejection

- (a) Draw $u \sim \mathcal{U}(0, 1)$.
 - (b) If $u \leq \alpha$, accept $(\mathcal{A}^*, \alpha^*, \mathcal{B}^*)$ and set $(\mathcal{A}_j, \alpha_j, \mathcal{B}_j) = (\mathcal{A}^*, \alpha^*, \mathcal{B}^*)$.
 - (c) If $u > \alpha$, reject $(\mathcal{A}^*, \alpha^*, \mathcal{B}^*)$ and set $(\mathcal{A}_j, \alpha_j, \mathcal{B}_j) = (\mathcal{A}_{j-1}, \alpha_{j-1}, \mathcal{B}_{j-1})$.
-

6.4.4 Tuning the Particle Filter

The particle filter, presented in Algorithm 6, includes two parameters that need to be tuned manually. These are the number of particles N and the standard deviation σ in the measurement model, defined by Equation (6.18). Moreover, also the duration T during which the complex reflectivity is measured needs to be chosen. Notably, T depends on the specific application and is not a parameter within the particle filter itself.

To find optimal parameters can be difficult. A larger T , results in a longer time series of the complex reflectivity, which means that more information about the sea clutter is collected but it also means a longer computational time. In addition, a larger T and no resampling, means that more weights approach zero and hence more particles are needed in order to avoid degeneracy. Overall, one wants to choose N and T as small as possible but at the same time letting T be large enough to contain enough information about the clutter and N large enough to avoid the degeneracy problem.

To find appropriate parameters, the particle filter (Algorithm 6) and the Metropolis-Hastings algorithm incorporating this PF (Algorithm 7) are evaluated for a wide range of different values. Mainly, it is controlled whether the Markov chains in Algorithm 7 seem to converge to the correct values for \mathcal{A} , α and \mathcal{B} . In addition, to find an appropriate σ , kernel density estimations (KDE) for different values of σ are compared with the distribution of the particles $\{\mathbf{y}_k^{(i)}\}_{i=1}^N$ at some times t_k of the time series.

The measurement model, $p(\Psi_k|\mathbf{y}_k) = \mathcal{N}(\mathbf{y}_k, \sigma^2)$ should describe the distribution of the particles $\{\mathbf{y}_k^{(i)}\}_{i=1}^N$ accurately. One way to verify this is by using KDE. It is a statistical technique used to estimate the probability density function of a continuous random variable based on a set of observed data points [29]. A kernel function is placed at each particle $\mathbf{y}_k^{(i)}$ and then these functions are summarized, resulting in a kernel density estimation. Since the measurement model is $p(\Psi_k|\mathbf{y}_k) = \mathcal{N}(\mathbf{y}_k, \sigma^2)$, the kernel function for each particle $\mathbf{y}_k^{(i)}$ becomes

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mathbf{y}_k^{(i)})}.$$

Here, x is the continuous interval from the lowest value of all the particles $\mathbf{y}_k^{(i)}$ to the highest value. This kernel density estimation can be compared to the histogram of the particles $\mathbf{y}_k^{(i)}$. If the KDE closely aligns with the histogram, it is assumed that the measurement model with variance σ^2 accurately describes the nonlinear state-space system. In Section 7.2.3, figures with the selected parameters N , T and σ are presented.

7

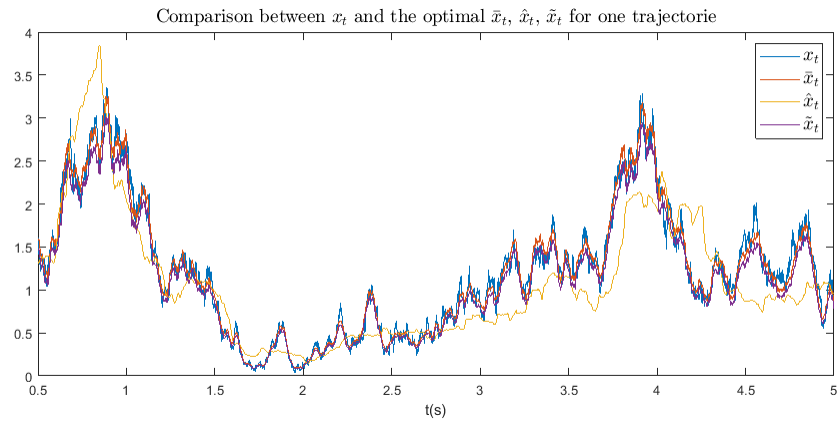
Results

This chapter aims to present the performance of the estimators and algorithms, proposed in Chapter 6. Section 7.1 presents the best estimators for the stochastic processes x_t (RCS) and γ_t (speckle). The selected estimators are then utilized as inputs to Algorithm 4 and 5. Simulation results from these algorithms are then presented in Section 7.2.1 and 7.2.2, respectively. Section 7.2.3 details the parameters chosen for the particle filter, presented in Algorithm 6, and provides figures that demonstrate their adequacy. Lastly, Section 7.2.4 shows the efficiency of Algorithm 7, designed for estimating the parameters \mathcal{A} , α and \mathcal{B} simultaneously by applying the particle filter for likelihood approximations. To evaluate the performance of Algorithm 4, 5 and 7, the mean value of the Markov chains, generated by the algorithms, are computed after the burn-in period.

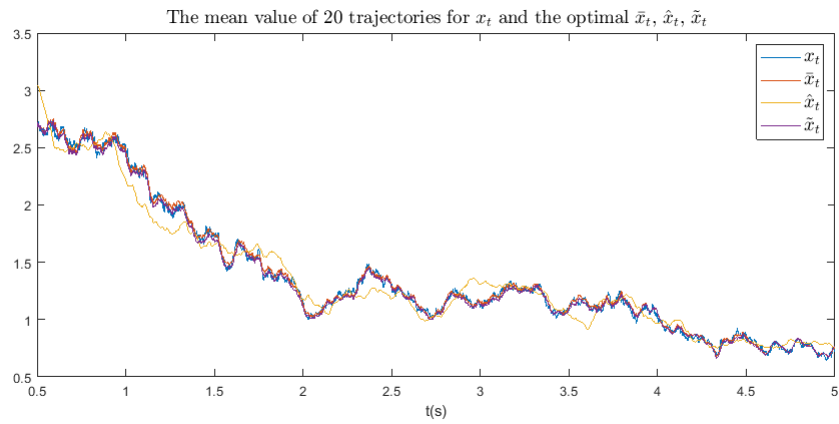
7.1 Estimators for x_t and γ_t

First, let x_t^* and γ_t^* denote the best estimators for x_t and γ_t , respectively. This section compares the three estimators for x_t , defined in Section 6.2 by Equations (6.3)-(6.5). Since only \hat{x}_t and \tilde{x}_t are estimators that are possible to obtain from real data, one of these are considered as the best estimator. Particularly, x_t^* is the estimator that most accurately follows the simulated x_t . This is evaluated by visually comparing sample paths of the estimators with the real simulated x_t and the corresponding root mean square errors (RMSEs). The stochastic process γ_t depends on x_t , and thus $\gamma_t^* = \Psi_t / \sqrt{x_t^*}$. The selected estimators are then utilized in Algorithm 5 and 4, respectively.

Figure 7.1 features a segment comprising 4.5 seconds out of a total of 300 seconds of the real simulated x_t and the three estimators \bar{x}_t , \hat{x}_t and \tilde{x}_t . Specifically, Figure 7.1a compares the estimators for a single sample path while Figure 7.1b illustrates the mean value obtained from 20 trajectories. Additionally, the corresponding RMSEs are illustrated in Figure 7.2. Remembering that only \hat{x}_t and \tilde{x}_t are possible estimators, visual inspection indicates that $x_t^* = \tilde{x}_t$, which is then utilized in Algorithm 5.



(a)



(b)

Figure 7.1: Comparison between x_t and the estimators $\bar{x}_t, \hat{x}_t, \tilde{x}_t$ for a segment comprising 4.5 seconds out of a total of 300 seconds. (a) Comparison for a single sample path. (b) Comparison of the average obtained from 20 trajectories.

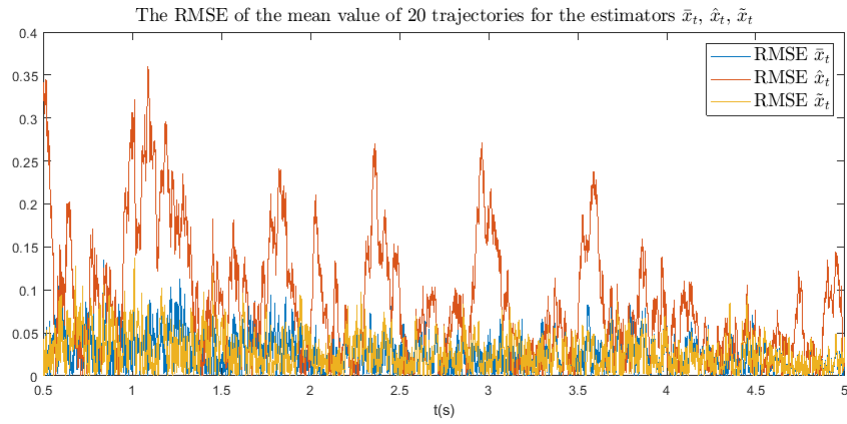
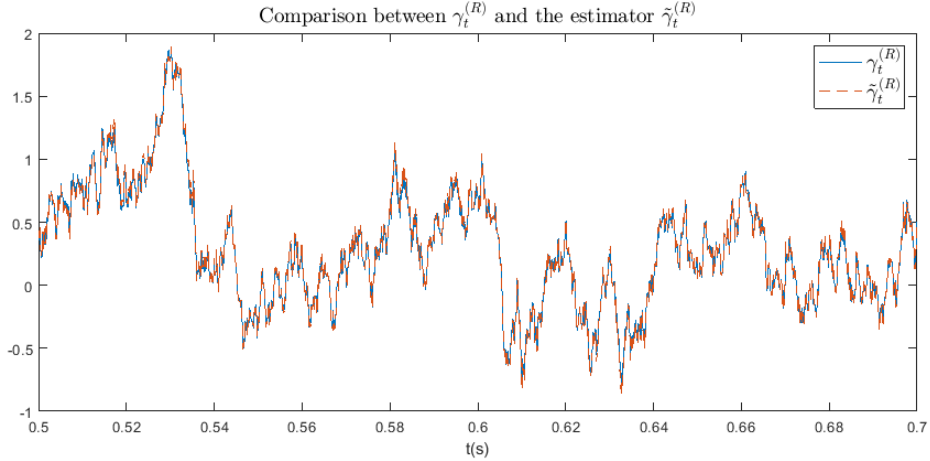


Figure 7.2: The RMSE for the three estimators over 4.5 seconds.

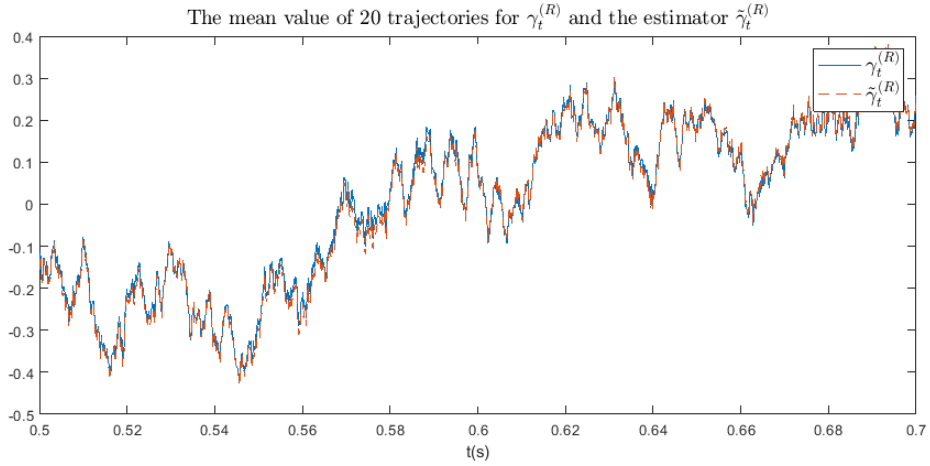
Since $x_t^* = \tilde{x}_t$, the selected estimator for γ_t is provided by

$$\gamma_t^* = \tilde{\gamma}_t = \frac{\Psi_t}{\tilde{x}_t^{1/2}}.$$

To illustrate its performance, only the real part of the estimator is considered, $\tilde{\gamma}_t^{(R)}$ (one can also equivalently consider the imaginary part $\tilde{\gamma}_t^{(I)}$). Figure 7.3 features a segment comprising 0.2 seconds out of a total of 300 seconds of the real simulated speckle $\gamma_t^{(R)}$ and the estimator $\tilde{\gamma}_t^{(R)}$. Specifically, Figure 7.3a illustrates the estimator for a single sample path while Figure 7.3b shows the average obtained from 20 trajectories. Additionally, the corresponding root mean square error is displayed in Figure 7.4. Indeed, this confirms that γ_t^* is a good estimator for γ_t . Thus, it is utilized in Algorithm 4.



(a)



(b)

Figure 7.3: Comparison between $\gamma_t^{(R)}$ and the estimator $\tilde{\gamma}_t^{(R)}$ for a segment comprising 0.2 seconds out of a total of 300 seconds. (a) Comparison for a single sample path. (b) Comparison of the average obtained from 20 trajectories.

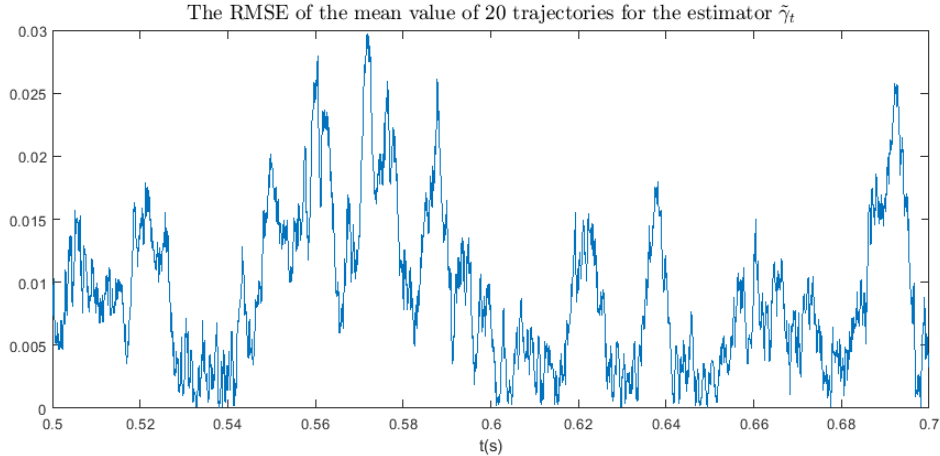


Figure 7.4: *The RMSE for $\tilde{\gamma}_t^{(R)}$ over 0.2 seconds.*

Figure 7.2 and 7.4 display the root mean square error for x_t^* and the real part of γ_t^* , respectively. These figures indicate that the RMSE for γ_t^* is notably lower than that of x_t^* . Thus, γ_t^* seems to be a better estimator than x_t^* . Since these estimators are utilized in Algorithm 4 and 5 for estimating the sea clutter parameters \mathcal{B} and \mathcal{A} , α , respectively, Algorithm 4 will probably perform better since it utilizes γ_t^* .

7.2 Bayesian Estimation Algorithms for Estimating Sea Clutter Parameters

In this section, the performance of the algorithms proposed in Section 6.3 and 6.4 are presented. To be able to evaluate the robustness and performance of the algorithms for various sea clutter data, two sets of parameters \mathcal{A} , α , \mathcal{B} are considered. Specifically, the observed complex reflectivity $\Psi_{1:n}$ is produced with two sets of parameters, see Table 7.1. Then, the Algorithms 4, 5 and 7 are run for the two sets.

Table 7.1: *Sets of parameter values from which the observed Reflectivity is produced.*

Set 1:	Set 2
$\mathcal{A} = 1$	$\mathcal{A} = 1$
$\alpha = 1$	$\alpha = 1$
$\mathcal{B} = 100$	$\mathcal{B} = 300$

First, in Section 7.2.1 the results regarding the estimation of \mathcal{B} using Metropolis-Hastings are presented. This is followed by a presentation of the performance of the Metropolis-Hasting algorithm for estimating \mathcal{A} and α in Section 7.2.2. Section 7.2.3 specifies the selected values for N , T and σ , which are used in the particle filter (see Algorithm 6). These values are then applied to the particle filter within Metropolis-Hastings algorithm. Results regarding the performance of this algorithm are then presented in Section 7.2.4.

7.2.1 Metropolis-Hastings for \mathcal{B}

To estimate the sea clutter parameter \mathcal{B} , one can use Algorithm 4 presented in Section 6.3.1. Ten runs are performed for each set of parameters defined in Table 7.1. The estimator $\tilde{\gamma}_t^{(R)}$ is used as data,

which serves as input to the algorithm. Moreover, instead of sampling from the prior distribution for \mathcal{B} , the same initial value \mathcal{B}_0 is used throughout all simulations. Variables used for producing the estimator $\tilde{\gamma}_t^{(R)}$, such as start values for the processes x_t , γ_t^R , γ_t^I (These are used in the simulation of the complex reflectivity Ψ_t), along with variables associated with the MH algorithm are compiled in Table 7.2.

Table 7.2: Variables used in the simulation process for estimating \mathcal{B} with Algorithm 4.

Estimator $\tilde{\gamma}_t^{(R)}$:					MH:	
x_0	γ_0^R	γ_0^I	Δt [s]	T [s]	M	\mathcal{B}_0
4	2	2	10^{-4}	300	$5 \cdot 10^3$	400

First the algorithm is run with the true parameter values $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. One representative simulation is presented in Figure 7.5. It includes the Markov chain of \mathcal{B} and the distribution of \mathcal{B} after the burn-in period. Approximately, the burn-in period constitutes the first 800 iterations for every simulation. However, as a precaution, the first 900 iterations are considered the burn-in. The same is done for the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. Although the burn-in period is shorter for this set of parameters, the first 900 iteration are again considered the burn-in, which makes the results comparable. One representative simulation for this set of parameters is presented in Figure 7.6.

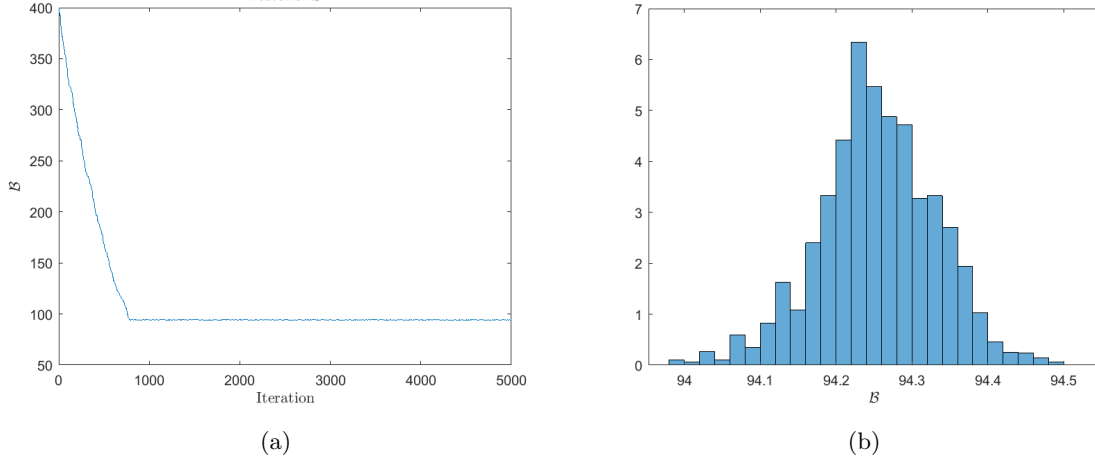


Figure 7.5: Markov chain and distribution of \mathcal{B} for one simulation with Algorithm 4. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{B} . (b) Distribution of \mathcal{B} after burn-in.

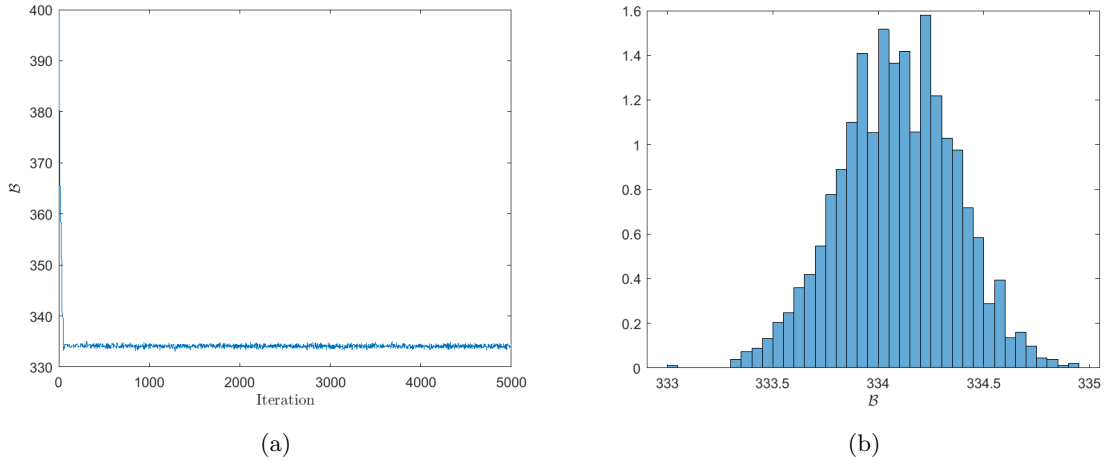


Figure 7.6: *Markov chain and distribution of \mathcal{B} for one simulation with Algorithm 4. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. (a) Markov chain for \mathcal{B} . (b) Distribution of \mathcal{B} after burn-in.*

The mean value of \mathcal{B} after the burn-in is presented in Table 7.3 for each simulation. By summarizing these values and then divide by 10, also an overall mean $\bar{\mathcal{B}}$, referred to as average, is obtained. It is observed in Figure 7.5 and 7.6, that Algorithm 4 generates Markov chains for \mathcal{B} that clearly converge towards distinct values. These values vary each simulation, presumably since a new time series of the complex reflectivity from which the estimated speckle $\tilde{\gamma}_t^{(R)}$ is produced, is generated each simulation. In addition, although it is shown in Section 6.2 that a time series of 300 seconds for the simulated reflectivity should be enough in order to produce an accurate estimator for $\gamma_t^{(R)}$, it may not be sufficient for all simulations. However, the averages of \mathcal{B} , calculated from 10 simulations, amount to $\bar{\mathcal{B}} = 99.1$ and $\bar{\mathcal{B}} = 300.9$, which are very close to the true values $\mathcal{B} = 100$ and $\mathcal{B} = 300$. Evidently, Algorithm 4 performs well when running it multiple times.

Table 7.3: *Estimated mean of the Markov chain for \mathcal{B} generated by Algorithm 4 after the burn-in for ten simulations and two different parameter sets.*

Real parameter values:	Estimated \mathcal{B} :										$\bar{\mathcal{B}}$ [Hz]
$\mathcal{A}=1, \alpha=1, \mathcal{B}=100$	94.5	104.7	115.5	104.1	78.6	85.9	112.5	101.9	98.9	94.3	99.1
$\mathcal{A}=1, \alpha=1, \mathcal{B}=300$	295.7	281.6	311.3	344.7	309.6	234.4	256.5	335.5	305.7	334.1	300.9

7.2.2 Metropolis-Hastings for \mathcal{A} and α

Here, the performance of Algorithm 5, designed for estimating \mathcal{A} and α with the Metropolis-Hastings method, is presented. As in the previous section, 10 runs are performed for each set of parameters defined in Table 7.1. The estimator \tilde{x}_t constitutes the data, which is input to the algorithm. Furthermore, instead of sampling from the prior distributions for \mathcal{A} and α , the same initial values \mathcal{A}_0 and α_0 are used throughout all simulations. Essential variables used for producing the estimator \tilde{x}_t together with variables associated with the MH algorithm are compiled in Table 7.4

Table 7.4: *Variables used in the simulation process for estimating \mathcal{A} and α with Algorithm 5.*

Estimator \tilde{x}_t :					MH:		
x_0	γ_0^R	γ_0^I	Δt [s]	T [s]	M	\mathcal{A}_0	α_0
4	2	2	10^{-4}	300	$15 \cdot 10^3$	3	4

The algorithm is first run with the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. One representative simulation is presented in Figure 7.7. Specifically, Figure 7.7a and 7.7c illustrate the Markov chains of \mathcal{A} and α , respectively. Moreover, Figure 7.7b and 7.7d show the distributions after burn-in of \mathcal{A} and α , respectively. The burn-in period is considered to constitute approximately the first 5000 iterations for each simulation. Resembling simulations are then performed with the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. To make the results comparable, the first 5000 iterations are again considered the burn-in. Figure 7.8 illustrates one representative simulation.

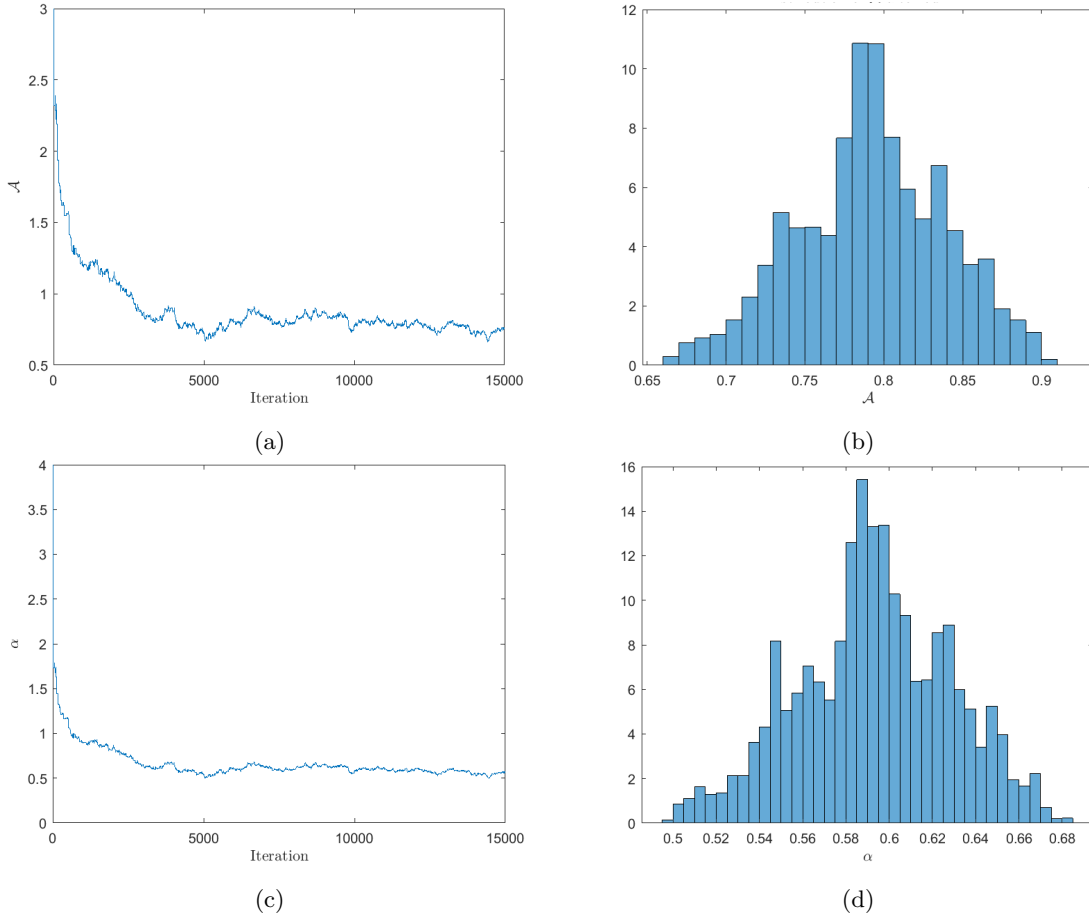


Figure 7.7: Markov chains and distributions of \mathcal{A} and α for one simulation with Algorithm 5. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{A} . (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in.

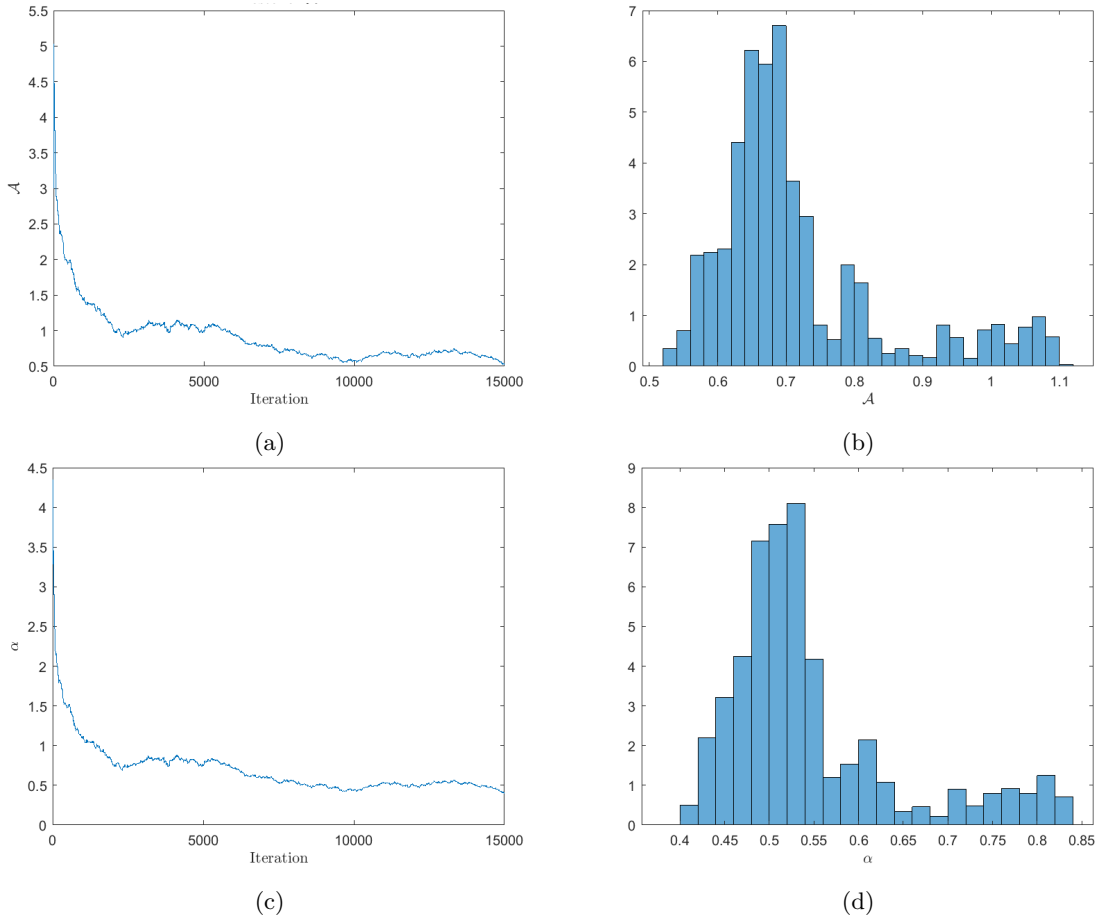


Figure 7.8: Markov chains and distributions of \mathcal{A} and α for one simulation with Algorithm 5. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. (a) Markov chain for \mathcal{A} . (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in.

The mean values for \mathcal{A} and α after burn-in are compiled in Table 7.5 together with the averages $\bar{\mathcal{A}}$ and $\bar{\alpha}$, obtained from ten simulations. In contrast to Algorithm 4, designed for estimating \mathcal{B} , Algorithm 5, does not yield as satisfying results. With the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$, the averages for \mathcal{A} and α , calculated from 10 simulations, are $\bar{\mathcal{A}} = 0.815$ and $\bar{\alpha} = 0.613$. Correspondingly, with the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$, the averages are $\bar{\mathcal{A}} = 0.756$ and $\bar{\alpha} = 0.561$. Again, it may be necessary to simulate the reflectivity more than 300 seconds. As anticipated in Section 7.1, Algorithm 4 performs better than Algorithm 5, probably since $\tilde{\gamma}_t^{(R)}$ is a better estimator than \tilde{x}_t .

Table 7.5: Estimated mean of the Markov chains for \mathcal{A} and α generated by Algorithm 5 after the burn-in for ten simulations and two different parameter sets.

Real parameter values:	Estimated \mathcal{A} :										$\bar{\mathcal{A}}$
$\mathcal{A}=1, \alpha=1, \mathcal{B}=100$	0.732	0.792	0.871	0.877	0.873	0.705	0.867	0.873	0.774	0.781	0.815
$\mathcal{A}=1, \alpha=1, \mathcal{B}=300$	0.692	0.871	0.602	0.662	0.620	0.891	0.896	0.672	0.932	0.717	0.756
Real parameter values:	Estimated α :										$\bar{\alpha}$
$\mathcal{A}=1, \alpha=1, \mathcal{B}=100$	0.544	0.593	0.659	0.680	0.658	0.497	0.639	0.676	0.580	0.599	0.613
$\mathcal{A}=1, \alpha=1, \mathcal{B}=300$	0.521	0.659	0.443	0.474	0.458	0.671	0.635	0.504	0.694	0.547	0.561

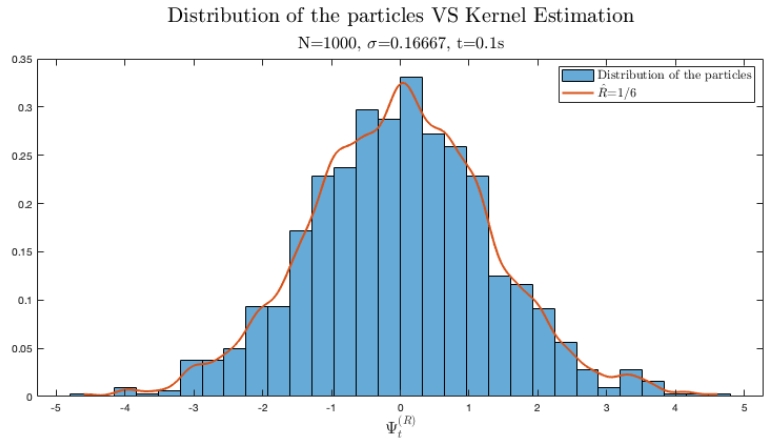
7.2.3 Selected N , T and σ for an Effective Particle Filter

As stated in Section 6.4.4, a wide range of values for N , T and σ are evaluated. The final values are presented in Table 7.6 below. These values serve as inputs for Algorithm 7 whose outcomes are detailed in the subsequent section.

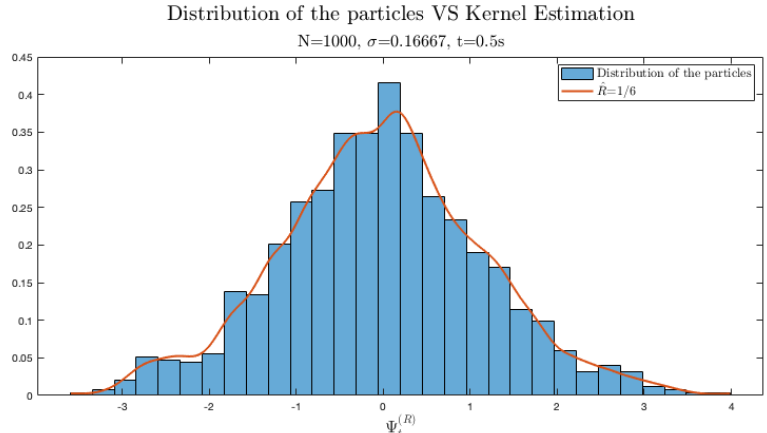
Table 7.6: *Values for N , T and σ .*

N	T [s]	σ
$1 \cdot 10^3$	3.0	1/6

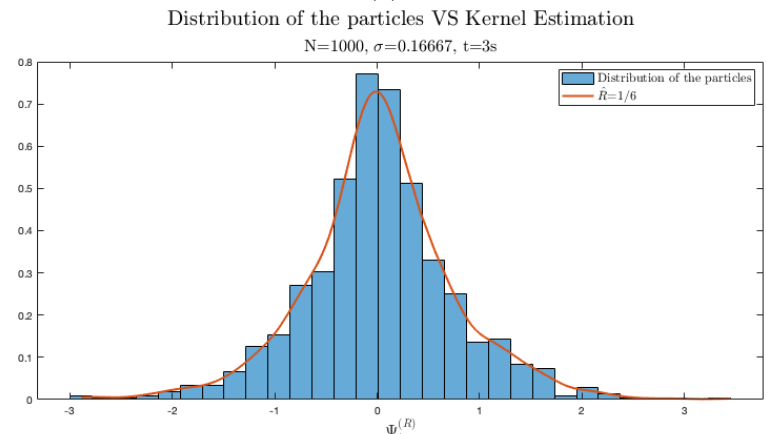
Figure 7.9 compares the kernel density estimation of the reflectivity $\Psi_t^{(R)}$ with the histogram of the particles $\{\mathbf{y}_t^{(i)}\}_{i=1}^N$, at three specific times $t = 0.1\text{s}$, $t = 0.5\text{s}$ and $T = 3\text{s}$. The figure illustrates that the kernel density estimations closely aligns with the particle distributions. Furthermore, Figure 7.10, illustrates how the reflectivity $\Psi^{(R)}$ evolves for the first 0.01s. Particularly, Figure 7.10a shows the trajectories for the real parts of the particles together with the real simulated reflectivity and Figure 7.10b shows the mean value of these trajectories, which is called the predicted mean, together with the real simulated reflectivity and the observed real reflectivity.



(a)



(b)



(c)

Figure 7.9: Kernel density estimations compared with particle distributions.

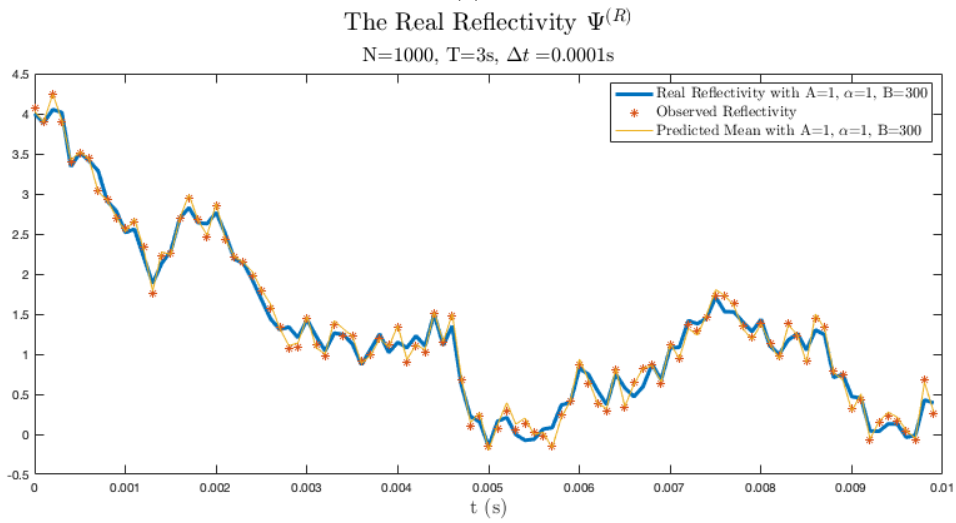
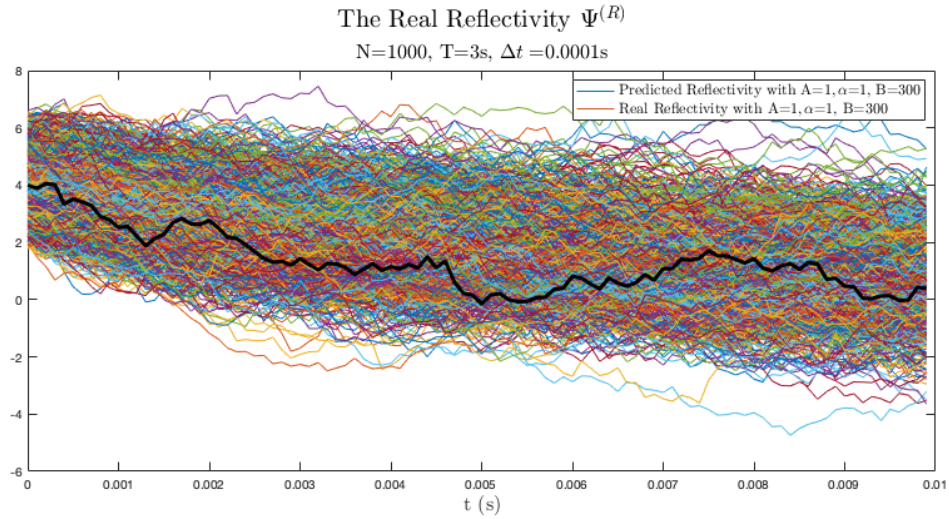


Figure 7.10: *The real part of the complex reflectivity $\Psi^{(R)}$.*

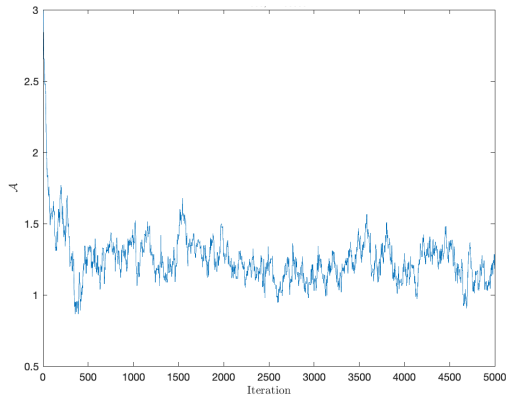
7.2.4 Particle Filter within Metropolis-Hastings for $\mathcal{A}, \alpha, \mathcal{B}$

This section illustrates the performance of Algorithm 7, designed for estimating the full posterior $p(\mathcal{A}, \alpha, \mathcal{B} | \Psi)$. Instead of sampling from the prior distributions for \mathcal{A} , α and \mathcal{B} , the same initial values \mathcal{A}_0 , α_0 and \mathcal{B}_0 are used throughout all simulations, as done in Section 7.2.1 and 7.2.2. Input values together with initial values \mathcal{A}_0 , α_0 and \mathcal{B}_0 and starting values for the stochastic processes in Fields model are compiled in Table 7.7. In contrast to the previously evaluated algorithms: Algorithm 4 and 5, Algorithm 7 utilizes a time series of the complex reflectivity of only 3 seconds.

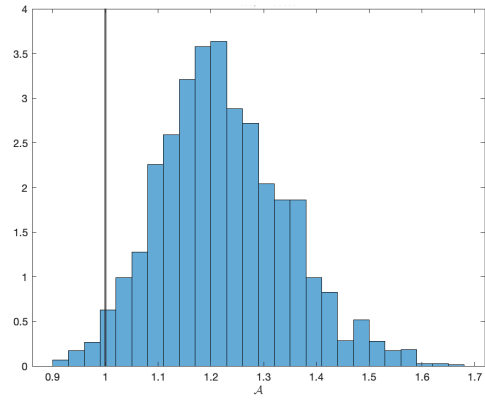
Table 7.7: *Initial values and variables serving as inputs to Algorithm 6 and 7.*

Reflectivity Ψ_t:					PF:		MH:			
x_0	$\gamma_0^{(R)}$	$\gamma_0^{(I)}$	Δt [s]	T [s]	N	σ	M	\mathcal{A}_0 [Hz]	α_0	\mathcal{B}_0 [Hz]
4	2	2	10^{-4}	3	10^3	1/6	$5 \cdot 10^3$	3	4	500

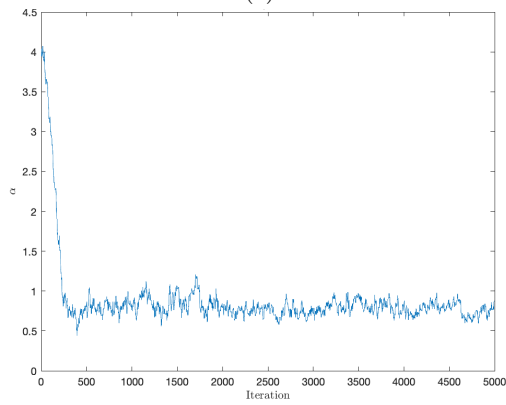
Since each simulation with Algorithm 7 takes approximately 5 hours only two simulations are performed for each set of true parameters. First, the algorithm is run with the true parameter values $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. One simulation is presented in Figure 7.11. The figure illustrates the Markov chains of each parameter \mathcal{A} , α and \mathcal{B} , and the distributions of each parameter after the burn-in period, which is considered the first 1000 iterations. Likewise, Figure 7.12 illustrates the same for the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. Figures for the other two simulations are found in Appendix C.



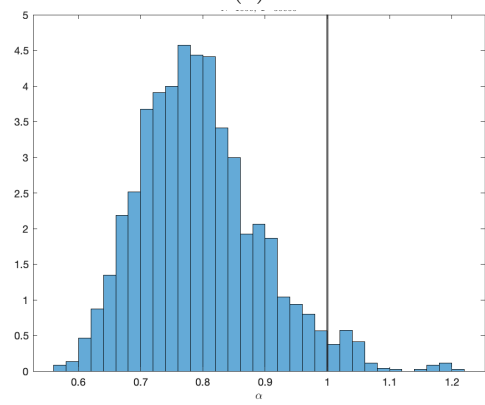
(a)



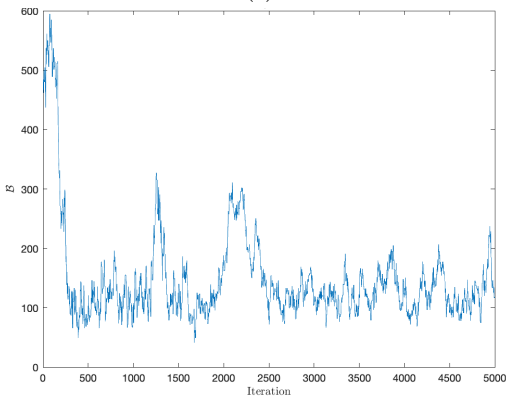
(b)



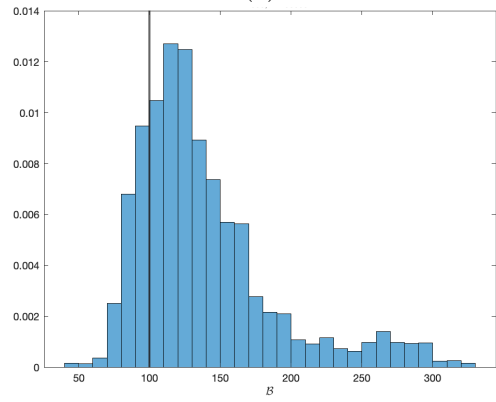
(c)



(d)



(e)



(f)

Figure 7.11: Markov chains and distributions of \mathcal{A} , α and \mathcal{B} for one simulation with Algorithm 7. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{A} . (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in. (e) Markov chain for \mathcal{B} . (f) Distribution of \mathcal{B} after burn-in.

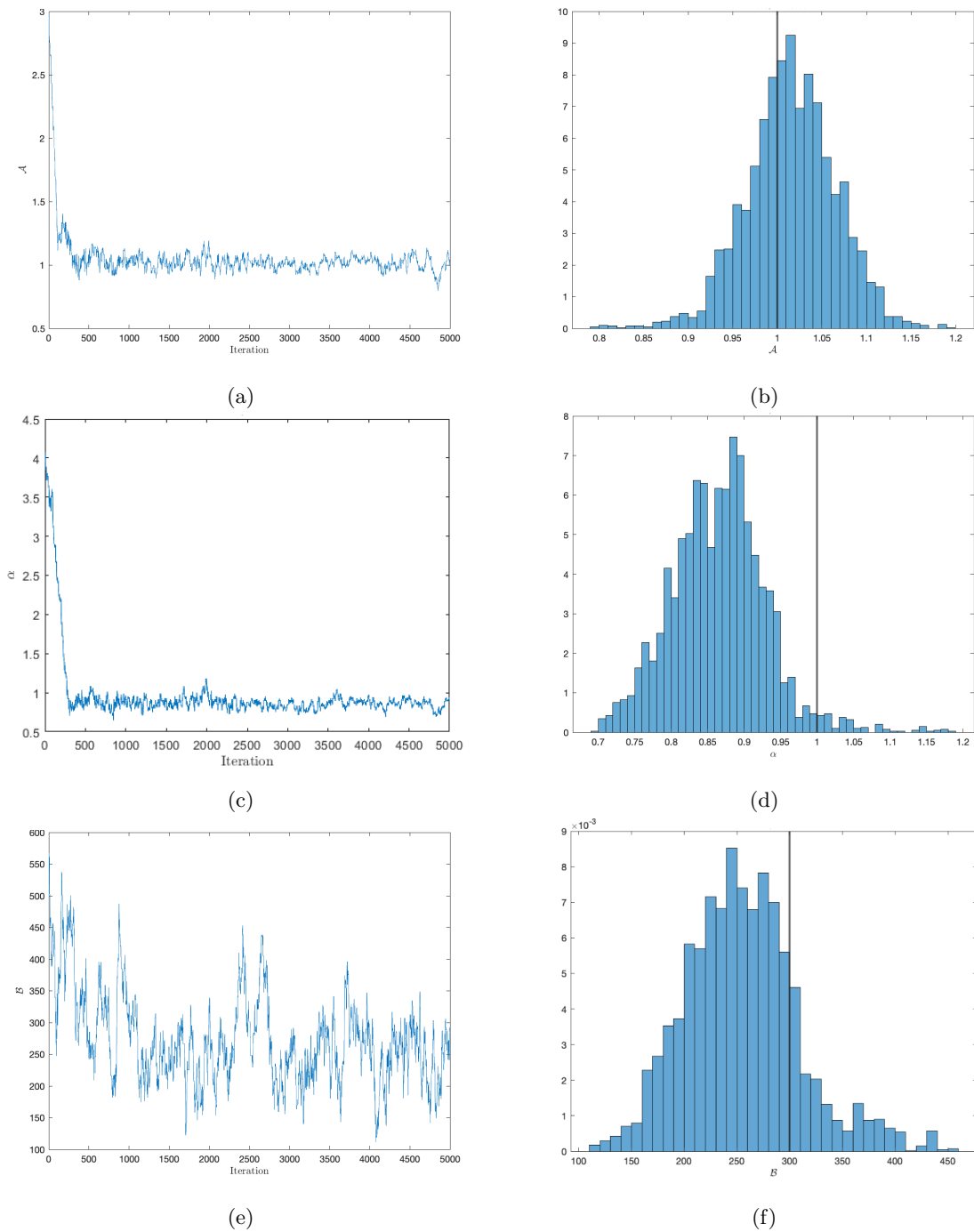


Figure 7.12: Markov chains and distributions of \mathcal{A} , α and \mathcal{B} for one simulation with Algorithm 7. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$. (a) Markov chain for \mathcal{A} . (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in. (e) Markov chain for \mathcal{B} . (f) Distribution of \mathcal{B} after burn-in.

The mean values for \mathcal{A} , α and \mathcal{B} after burn-in are compiled in Table 7.8 and 7.9 together with the estimated averages $\bar{\mathcal{A}}$, $\bar{\alpha}$ and $\bar{\mathcal{B}}$, obtained from two simulations. All Markov chains indicate convergence

towards values that are relatively close to the true values. For the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$, the averages, calculated from 2 simulations, are $\bar{\mathcal{A}} = 1.137$, $\bar{\alpha} = 0.881$ and $\bar{\mathcal{B}} = 118.3$. Similarly, for the true parameters $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 300$, the averages amount to $\bar{\mathcal{A}} = 1.174$, $\bar{\alpha} = 0.959$ and $\bar{\mathcal{B}} = 209.4$. This algorithm exhibits a fairly robust performance as it produces estimated values that closely resemble the actual values. Nevertheless, to gain a more certain understanding of the algorithm's effectiveness, it would be necessary to run it multiple times.

Table 7.8: *Estimated mean of the Markov chains for \mathcal{A} , α and \mathcal{B} generated by Algorithm 7 after the burn-in for two simulations and set 1 of true parameters.*

Parameter	Real value	Estimated mean	Average
\mathcal{A} [Hz]	1	1.224 1.050	1.137
α	1	0.795 0.966	0.881
\mathcal{B} [Hz]	100	138.4 98.29	118.3

Table 7.9: *Estimated mean of the Markov chains for \mathcal{A} , α and \mathcal{B} generated by Algorithm 7 after the burn-in for two simulations and set 2 of true parameters.*

Parameter	Real value	Estimated mean	Estimated variance
\mathcal{A} [Hz]	1	1.015 1.332	1.174
α	1	0.865 1.053	0.959
\mathcal{B} [Hz]	300	254.8 163.9	209.4

Future Work

The provided algorithms make a valuable contribution to the understanding of the statistical characteristics of sea clutter. In particular, they provide two Bayesian approaches for estimating the sea clutter parameters within Field's model. Future work will have to deal with the issues of devising faster methods to achieve accurate estimations. Additionally, future work should aim to apply these techniques to real-world data. The proposed ideas for future work (listed below), mainly applies to Algorithm 7.

1. For a more reliable assessment of the performance of Algorithm 7, it is imperative to execute the algorithm multiple times. Initially, it would be of interest to perform more simulations for the two sets of parameters considered in this work. Subsequently, to achieve a comprehensive assessment of the algorithm, it is necessary to evaluate the algorithm for a diverse range of sea clutter data.
2. The particle filter designed for likelihood approximation (Algorithm 6) does not include a resampling step. The incorporation of an appropriate resampling step, which mitigates the degeneracy problem while maintaining diversity among particles, could potentially improve the performance of Algorithm 7.
3. In this project, synthetic data has been employed. It would be interesting to repeat the simulations and experiments using real data, that is, data obtained by illuminating a sea surface with a radar. For example, the IPIX data, measured with a fully coherent X-band radar with advanced features, could be used [30]. This approach would provide a real-world evaluation of Field's model and the methods of Bayesian Inference used in this work.
4. To enable parameter estimation in real time, one needs to find faster methods. The current particle filter, employed in Algorithm 7 for likelihood approximation, is computational extensive. Instead of utilizing a particle filter for this task, it may be possible to employ a neural network for likelihood approximation. Potentially, this could improve the efficiency of Algorithm 7.
5. Both the complex Ornstein-Uhlenbeck and the Cox-Ingersson-Ross processes are comprehensively studied processes, exhibiting rich structures. Using, for instance, the fact that the OU process is linear, one should be able to make a hybrid Kalman/Particle filter for likelihood approximation. This would result in faster estimation, since the three dimensional particle filter (Algorithm 6) would be replaced with a more efficient one dimensional particle filter.
6. It could be of interest to use numerical methods of higher order for the discretization of the stochastic differential equations constituting Field's model. Applying methods, such as the higher order Taylor schemes described on pages 146-150 in [31], could improve the accuracy of the suggested algorithms.

Conclusion

The aim of this work was to explore the possibilities of estimating the parameters \mathcal{A} , α and \mathcal{B} in Field's model through the application of Bayesian inference. To achieve this objective, two distinct approaches were employed: one necessitating a time series of 300 seconds for the complex reflectivity Ψ_t , and another requiring only 3 seconds. The former approach is implemented in two Metropolis-Hastings algorithms: the first estimates \mathcal{B} and the second estimates \mathcal{A}, α . The latter approach is applied in a Metropolis-Hastings algorithm that estimates the three parameters simultaneously by employing a particle filter for likelihood approximations.

This study highlights the efficiency of Bayesian techniques, specifically the Metropolis-Hastings and Sequential Monte Carlo methods, in accurately estimating sea clutter parameters. Both the MH algorithm designed for estimating \mathcal{B} and the Metropolis-Hastings algorithm, incorporating a particle filter for likelihood approximations, demonstrate strong performance. Nonetheless, it is important to consider the applicability of these methods for diverse sea clutter returns, meaning clutter data produced with a wide range of sea clutter parameters. Additionally, their computational intensity presents challenges for real-world applications. Future research should address the need for more computationally efficient methods to overcome these challenges.

Bibliography

- [1] Clement J. Roussel. *Stochastic differential equations for the electromagnetic field scattered by the sea surface: applications to remote sensing*. HAL, 2020.
- [2] Merrill I. Skolnik. *Radar Handbook, 3rd Edition*. McGraw-Hill, 2008.
- [3] Keith D. Ward, Robert J. A. Tough, and Simon Watts. *Sea Clutter: Scattering, the K Distribution and Radar Performance*. The Institution of Engineering and Technology, London, United Kingdom, 2006.
- [4] Timothy R. Field. *Electromagnetic Scattering from Random Media*. Oxford University Press, 2009.
- [5] Clement Julien Roussel, Arnaud Coatanhay, and Alexandre Baussard. *Estimation of the parameters of stochastic differential equations for sea clutter*. The Institution of Engineering and Technology, 2018.
- [6] Mark A. Richards, James A. Scheer, and William A. Holm. *Principles of Modern Radar, Volume 1: Basic Principles*. Stevenage: The Institution of Engineering and Technology, 2010.
- [7] Stanislaw Rosloniec. *Fundamentals of the Radiolocation and Radionavigation*. Cham, Switzerland: Springer International Publishing, 2023.
- [8] Clive Alabaster. *Pulse Doppler Radar principles, technology, applications*. Edison, NJ : SciTech Publishing, 2012.
- [9] Dr. Robert M. O'Donnell. *Introduction to Radar Systems: Radar Equation - Online Course*. MIT Lincoln Laboratory. URL: <https://www.ll.mit.edu/outreach/radar-introduction-radar-systems-online-course>.
- [10] Luke Rosenberg and Simon Watts. *Radar Sea Clutter: Modelling and Target Detection*. Stevenage: Institution of Engineering and Technology, 2022.
- [11] Desmond J. Higham and Peter E. Kloeden. *An Introduction to the Numerical Simulation of Stochastic Differential Equations*. Philadelphia : Society for Industrial and Applied Mathematics, 2021. Chap. 5: Stochastic Differential Equations, pp. 51–62.
- [12] Desmond J. Higham. *An Algorithmic Introduction to Numerical Simulation of Stochastic Differential Equations*. Society for Industrial and Applied Mathematics, 2001.
- [13] Andrei N. Borodin. *Stochastic Processes*. Cham: Springer International Publishing: Imprint: Birkhäuser, 2017.
- [14] Hannes Risken. *The Fokker-Planck Equation: Methods of Solution and Applications*. Springer Berlin Heidelberg, 1984.
- [15] Ross A. Maller, Gernot Müller, and Alex Szimayer. *Handbook of Financial Time Series*. Springer Berlin Heidelberg, 2009. Chap. Ornstein-Uhlenbeck Processes and Extensions, pp. 421–437.
- [16] Xuerong Mao. *Stochastic Differential Equations and Applications*. 2nd ed. Cambridge, England: Woodhead Publishing, 2011.

- [17] Damiano Brigo and Fabio Mercurio. *Interest Rate Models Theory and Practice*. Springer Berlin Heidelberg: Imprint: Springer, 2001.
- [18] Peter E. Kloeden and Echarad Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 2013.
- [19] Dirk P. Kroese, Thomas Taimre, and Zdravko I. Botev. *Handbook of Monte Carlo Methods*. John Wiley & Sons, 2011.
- [20] Ming-Hui Chen, Qi-Man Shao, and Joseph G. Ibrahim. *Monte Carlo Methods in Bayesian Computation*. Springer, 2000.
- [21] Heikki Haario, Eero Saksman, and Johanna Tamminen. *Adaptive proposal distribution for random walk Metropolis algorithm*. Physica-Verlag, 1999.
- [22] Heikki Haario, Eero Saksman, and Johanna Tamminen. *An Adaptive Metropolis Algorithm*. Bernoulli, 2001.
- [23] Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.
- [24] Simo Sarkka. *Bayesian Filtering and Smoothing*. West Nyack: Cambridge University Press, 2013.
- [25] Anton J. Haug. *Bayesian estimation and tracking: a practical guide*. Newark: WILEY, 2012.
- [26] Marcelo G.S. Bruno. *Sequential Monte Carlo Methods for Nonlinear Discrete-Time Filtering*. Netherlands: Springer Nature, 2022.
- [27] M.Saneev Arulampalam et al. *A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking*. Institute of Electrical and Electronics Engineers, 2002.
- [28] T. Nohara and S. Haykin. *Canadian East Coast radar trials and the K-distribution*. IEE Proceedings F (Radar and Signal Processing). HERTFORD: Inst Engineering Technology-Iet, 1991.
- [29] Artur Gramacki. *Nonparametric Kernel Density Estimation and Its Computational Aspects*. Cham: Springer International Publishing: Imprint: Springer, 2018.
- [30] Rembrandt Bakker and Brian Currie. *The mcmaster ipix radar sea clutter*. 2001. URL: <http://soma.ece.mcmaster.ca/ipix/>.
- [31] Peter E. Kloeden, Echarad Platen, and Henri Schurz. *Numerical solution of SDE through computer experiments*. Berlin, Heidelberg: Springer, 1994.

A

Algorithms

A.1 Sequential Importance Resampling

Algorithm 8 Sequential Importance Resampling (SIR)

Input: $\mathbf{z}_{1:T}, N$

Output: $\{\mathbf{x}_{0:T}^{(i)}\}_{i=1}^N, \{\tilde{w}_{0:T}^{(i)}\}_{i=1}^N$

1. **Initialization**, $k=0$

- (a) Draw N particles from the the initial prior distribution

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \quad i = 1, \dots, N.$$

- (b) Set weights uniformly

$$\tilde{w}_0^{(i)} = \frac{1}{N}, \quad i = 1, \dots, N.$$

2. **For** $k = 1 : T$

- (a) Propagate particles according to the importance distribution

$$\mathbf{x}_k^{(i)} \sim q(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_{1:k}), \quad i = 1, \dots, N.$$

- (b) Compute the weights $w_k^{(i)}$ according to equation (4.10).

- (c) Compute the normalized weights

$$\tilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^N w_k^{(j)}}, \quad i = 1, \dots, N.$$

- (d) Compute the Effective number of particles according to

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^N \left(w_k^{(i)}\right)^2}.$$

- (e) If \hat{N}_{eff} is less than some threshold N_{thr} , then perform resampling.

- (f) Reset the weights uniformly

$$w_k^{(i)} = \frac{1}{N}, \quad i = 1, \dots, N.$$

A.2 The Bootstrap Filter

Algorithm 9 The Bootstrap Filter

Input: $\mathbf{z}_{1:T}, N$

Output: $\{\mathbf{x}_{0:T}^{(i)}\}_{i=1}^N, \{\tilde{w}_{0:T}^{(i)}\}_{i=1}^N$

1. **Initialization**, $k=0$

(a) Draw N particles from the the initial prior distribution

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \quad i = 1, \dots, N.$$

(b) Set weights uniformly

$$\tilde{w}_0^{(i)} = \frac{1}{N}, \quad i = 1, \dots, N.$$

2. **For** $k = 1 : T$

(a) Propagate particles according to the state-transition model

$$\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}), \quad i = 1, \dots, N.$$

(b) Calculate the conditional likelihood of observing the measurement \mathbf{y}_k given the predictions $\mathbf{x}_k^{(i)}$

$$p(\mathbf{y}_k | \mathbf{x}_k^{(i)}), \quad i = 1, \dots, N.$$

(c) Compute the unnormalized weights according to equation (4.12)

$$w_k^{(i)} = \tilde{w}_{k-1}^{(i)} p(\mathbf{y}_k | \mathbf{x}_k^{(i)}), \quad i = 1, \dots, N.$$

(d) Compute the normalized weights

$$\tilde{w}_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^N w_k^{(j)}}, \quad i = 1, \dots, N.$$

(e) Compute the Effective number of particles according to some criteria

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^N \left(w_k^{(i)}\right)^2}.$$

(f) If \hat{N}_{eff} is less than some threshold N_{thr} , then perform resampling.

(g) Reset the weights uniformly

$$w_k^{(i)} = \frac{1}{N}, \quad i = 1, \dots, N.$$

B

Derivations

This section summarizes the derivations of the three estimators \bar{x} , \hat{x} , and \tilde{x} , provided by Roussel in his work [1].

B.1 Estimator \bar{x}_t

The estimator \bar{x}_t is given by

$$\bar{x}_t = \frac{1}{\mathcal{B}\Delta N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2.$$

Derivation

First, let the increments of the complex reflectivity be denoted as

$$\Delta_{t_k} \Psi = \Psi_{t_k} - \Psi_{t_{k-1}}. \quad (\text{B.1})$$

Equation (B.1) can be rewritten according to

$$\begin{aligned} \Delta_{t_k} \Psi &= \Psi_{t_k} - \Psi_{t_{k-1}} \\ &= x_{t_k}^{1/2} \gamma_{t_k} - x_{t_{k-1}}^{1/2} \gamma_{t_{k-1}} \\ &= \left(x_{t_k}^{1/2} \gamma_{t_k}^{(R)} - x_{t_{k-1}}^{1/2} \gamma_{t_{k-1}}^{(R)} \right) + i \left(x_{t_k}^{1/2} \gamma_{t_k}^{(I)} - x_{t_{k-1}}^{1/2} \gamma_{t_{k-1}}^{(I)} \right). \end{aligned}$$

This yields

$$|\Delta_{t_k} \Psi|^2 = \left(x_{t_k}^{1/2} \gamma_{t_k}^{(R)} - x_{t_{k-1}}^{1/2} \gamma_{t_{k-1}}^{(R)} \right)^2 + \left(x_{t_k}^{1/2} \gamma_{t_k}^{(I)} - x_{t_{k-1}}^{1/2} \gamma_{t_{k-1}}^{(I)} \right)^2. \quad (\text{B.2})$$

If $\Delta t = t_k - t_{k-1}$ is small compared to the timescale of x_t , which for example is true for $\mathcal{A} = 1$ and $\Delta t = 10^{-4}$, then $x_{t_{k-1}} \approx x_{t_k}$ and Equation (B.2) can be rewritten as

$$\begin{aligned} |\Delta_{t_k} \Psi|^2 &\approx x_{t_k} \left[\left(\gamma_{t_k}^{(R)} - \gamma_{t_{k-1}}^{(R)} \right)^2 + \left(\gamma_{t_k}^{(I)} - \gamma_{t_{k-1}}^{(I)} \right)^2 \right] \\ &= x_{t_k} \left[\left(\Delta_k \gamma^{(R)} \right)^2 + \left(\Delta_k \gamma^{(I)} \right)^2 \right]. \end{aligned} \quad (\text{B.3})$$

The Euler-Maruyama scheme is applied to the stochastic differential equations for $\gamma_{t_k}^{(R)}$ and $\gamma_{t_k}^{(I)}$ (see Equations (6.1) and (6.2)). By using the fact that Δt is small, which means that the volatility term dominates the drift term in the Euler-Maruyama schemes, one can write

$$\begin{cases} \Delta_k \gamma^{(R)} \approx \frac{1}{\sqrt{2}} \mathcal{B}^{1/2} \Delta W_k^{(R)} = \frac{1}{\sqrt{2}} \mathcal{B}^{1/2} n_k^{(R)} \Delta t^{1/2}, \\ \Delta_k \gamma^{(I)} \approx \frac{1}{\sqrt{2}} \mathcal{B}^{1/2} \Delta W_k^{(I)} = \frac{1}{\sqrt{2}} \mathcal{B}^{1/2} n_k^{(I)} \Delta t^{1/2}, \end{cases} \quad (\text{B.4})$$

where $n_k^{(R)}$ and $n_k^{(I)}$ are independent normal random variables such that for all k , $n_k^{(R)} \sim \mathcal{N}(0, 1)$ and $n_k^{(I)} \sim \mathcal{N}(0, 1)$. Equation (B.4) is applied to Equation (B.3), which yields

$$\begin{aligned} |\Delta_{t_k} \Psi|^2 &= x_{t_k} \left[\left(\frac{1}{\sqrt{2}} \mathcal{B}^{1/2} n_k^{(R)} \Delta t^{1/2} \right)^2 + \left(\frac{1}{\sqrt{2}} \mathcal{B}^{1/2} n_k^{(I)} \Delta t^{1/2} \right)^2 \right] \\ &= x_{t_k} \left[\frac{\mathcal{B}}{2} n_k^{(R)2} \Delta t + \frac{\mathcal{B}}{2} n_k^{(I)2} \Delta t \right] \\ &= x_{t_k} \frac{\mathcal{B} \Delta t}{2} \left(n_k^{(R)2} + n_k^{(I)2} \right). \end{aligned}$$

To estimate x_t , a time window Δ_t centered at t with N elements is utilized. It is assumed that for all k such that $t_k \in \Delta_t$, one have $x_{t_k} \approx x_t$. Then, the squared increments of Ψ_t is averaged over the time window Δ_t . Subsequently, ergodicity and the fact that $\mathbb{E} \left[n_k^{(R)2} \right] = \mathbb{E} \left[n_k^{(I)2} \right] = 1$ yield

$$\begin{aligned} \frac{1}{N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2 &\approx x_t \frac{\mathcal{B} \Delta t}{2} \left(\frac{1}{N} \sum_{t_k \in \Delta_t} n_k^{(R)2} + \frac{1}{N} \sum_{t_k \in \Delta_t} n_k^{(I)2} \right) \\ &\approx \mathcal{B} \Delta t x_t. \end{aligned}$$

Then, an estimator for x_t is given by

$$\bar{x}_t = \frac{1}{\mathcal{B} \Delta N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2.$$

B.2 Estimator \hat{x}_t

The estimator \hat{x}_t is given by

$$\hat{x}_t = \frac{1}{N} \sum_{t_k \in \Delta_t} z_{t_k},$$

recalling that z_{t_k} represents the intensity.

Derivation

Remember from Section 5.3 that $\gamma_t^{(R)}$ and $\gamma_t^{(I)}$ are centered Gaussian distributed according to $\gamma_t^{(R)} \sim \mathcal{N}(0, \frac{1}{2})$, $\gamma_t^{(I)} \sim \mathcal{N}(0, \frac{1}{2})$, respectively. Then, the following holds true

$$\begin{aligned} \mathbb{E} [|\gamma_t|^2] &= \mathbb{E} \left[|\gamma_t^{(R)} + i\gamma_t^{(I)}|^2 \right] = \mathbb{E} \left[\gamma_t^{(R)2} + \gamma_t^{(I)2} \right] = 2\mathbb{E} \left[\gamma_t^{(R)2} \right] \\ &= 2 \left(\text{Var} \left[\gamma_t^{(R)} \right] + \mathbb{E} \left[\gamma_t^{(R)} \right]^2 \right) = 2 \left(\frac{1}{2} + 0 \right) = 1. \end{aligned} \tag{B.5}$$

Let Δ_t denote a time window centered at t . If the size of the window Δ_t is small compared to the decorrelation time of x_t (approximately $1/\mathcal{A}$) and large compared to the decorrelation time of γ_t (approximately $1/\mathcal{B}$), then $x_{t_k} \approx x_t$ for all $t_k \in \Delta_t$. Ergodicity and Equation (B.5), yields

$$\frac{1}{N} \sum_{t_k \in \Delta_t} |\gamma_{t_k}|^2 \approx 1,$$

where N is the number of elements in the time window Δ_t . Subsequently, one can write

$$\frac{1}{N} \sum_{t_k \in \Delta_t} z_{t_k} = \frac{1}{N} \sum_{t_k \in \Delta_t} x_{t_k} |\gamma_{t_k}|^2 \approx x_t \frac{1}{N} \sum_{t_k \in \Delta_t} |\gamma_{t_k}|^2 \approx x_t.$$

Thus, an estimator for x_t is given by

$$\hat{x}_t = \frac{1}{N} \sum_{t_k \in \Delta_t} z_{t_k}.$$

B.3 Estimator \tilde{x}_t

The estimator \tilde{x}_t is given by

$$\tilde{x}_t = \frac{1}{\tilde{\mathcal{B}}_\Psi \Delta N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2.$$

Derivation

Consider the estimator \bar{x} , derived in Appendix B.1

$$\bar{x}_t = \frac{1}{\mathcal{B} \Delta N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2. \quad (\text{B.6})$$

The estimator \tilde{x}_t is almost identical to \bar{x}_t with the difference that it uses an estimator for \mathcal{B} . Remember from Section 5.3 that x_t is Gamma distributed according to $x_t \sim \Gamma(\alpha, \alpha)$. Then, one can expect that the following holds true

$$\mathbb{E}[\bar{x}_t] \approx \mathbb{E}[x_t] = \frac{\alpha}{\alpha} = 1.$$

Let \bar{X}_t denote the following reformulation of Equation (B.6)

$$\bar{X}_t = \frac{1}{\Delta N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2 = \mathcal{B} \bar{x}_t.$$

Thus,

$$\mathbb{E}[\bar{X}_t] = \mathbb{E} \left[\frac{1}{\Delta N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2 \right] = \mathcal{B} \mathbb{E}[\bar{x}_t] \approx \mathcal{B}.$$

It is assumed that \bar{X}_t is ergodic and that \bar{X}_t is computed at times t_i for $i = 1, 2, \dots, m$ by centering the averaging window successively over t_i . This yields

$$\mathbb{E}[\bar{X}_t] = \lim_{T \rightarrow +\infty} \frac{1}{T} \int_0^T \bar{X}_r dr \approx \frac{1}{m \Delta t} \sum_{i=1}^m \bar{X}_{t_i} \Delta t = \frac{1}{m} \sum_{i=1}^m \bar{X}_{t_i}$$

Thus, the following estimator for \mathcal{B} is obtained

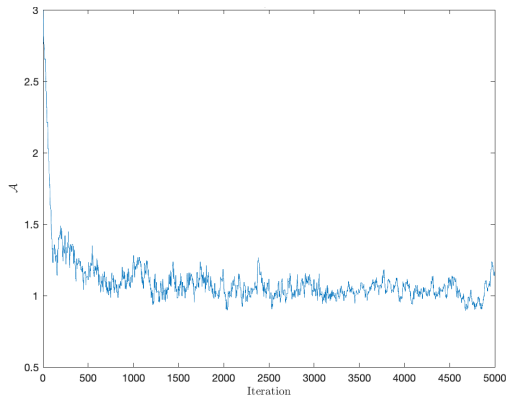
$$\tilde{B}_\Psi = \frac{1}{m} \sum_{i=1}^m \bar{X}_{t_i} = \frac{1}{m \Delta t N} \sum_{i=1}^m \sum_{t_k \in \Delta_i} |\Delta_{t_k} \Psi|^2.$$

Here Δ_i is a window centered at time t_i . Consequently, an estimator for x_t is given by

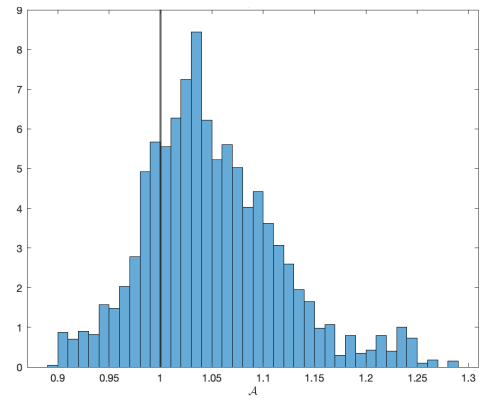
$$\bar{x}_t = \frac{1}{\tilde{\mathcal{B}}_\Psi \Delta N} \sum_{t_k \in \Delta_t} |\Delta_{t_k} \Psi|^2.$$

C

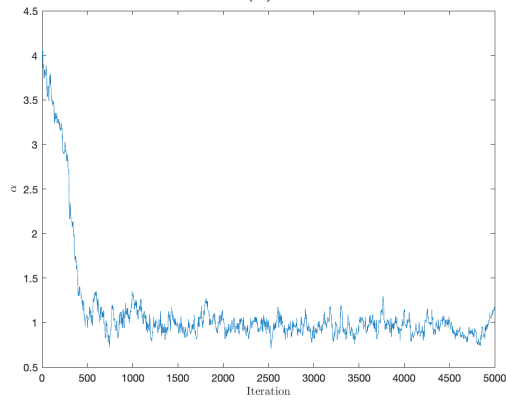
Figures



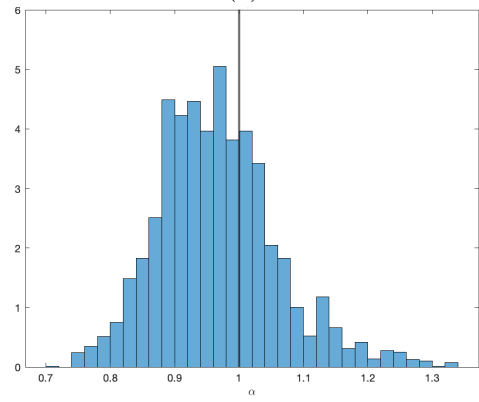
(a)



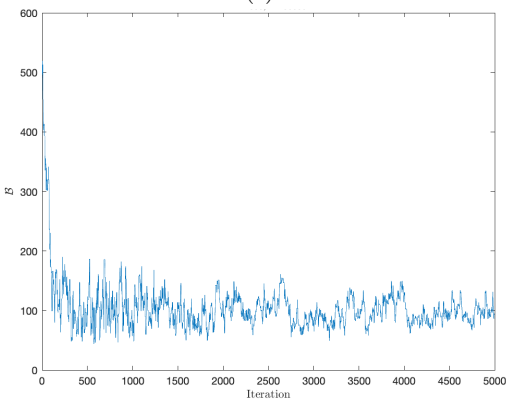
(b)



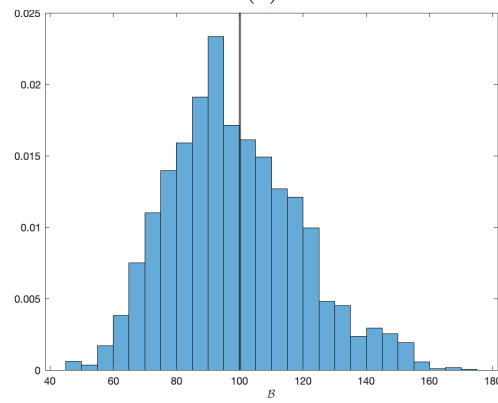
(c)



(d)



(e)



(f)

Figure C.1: (*Simulation 2*) Markov chains and distributions of \mathcal{A} , α and \mathcal{B} for one simulation with Algorithm 7. The true parameter values are $\mathcal{A} = 1$, $\alpha = 1$ and $\mathcal{B} = 100$. (a) Markov chain for \mathcal{A} . (b) Distribution of \mathcal{A} after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in. (e) Markov chain for \mathcal{B} . (f) Distribution of \mathcal{B} after burn-in.

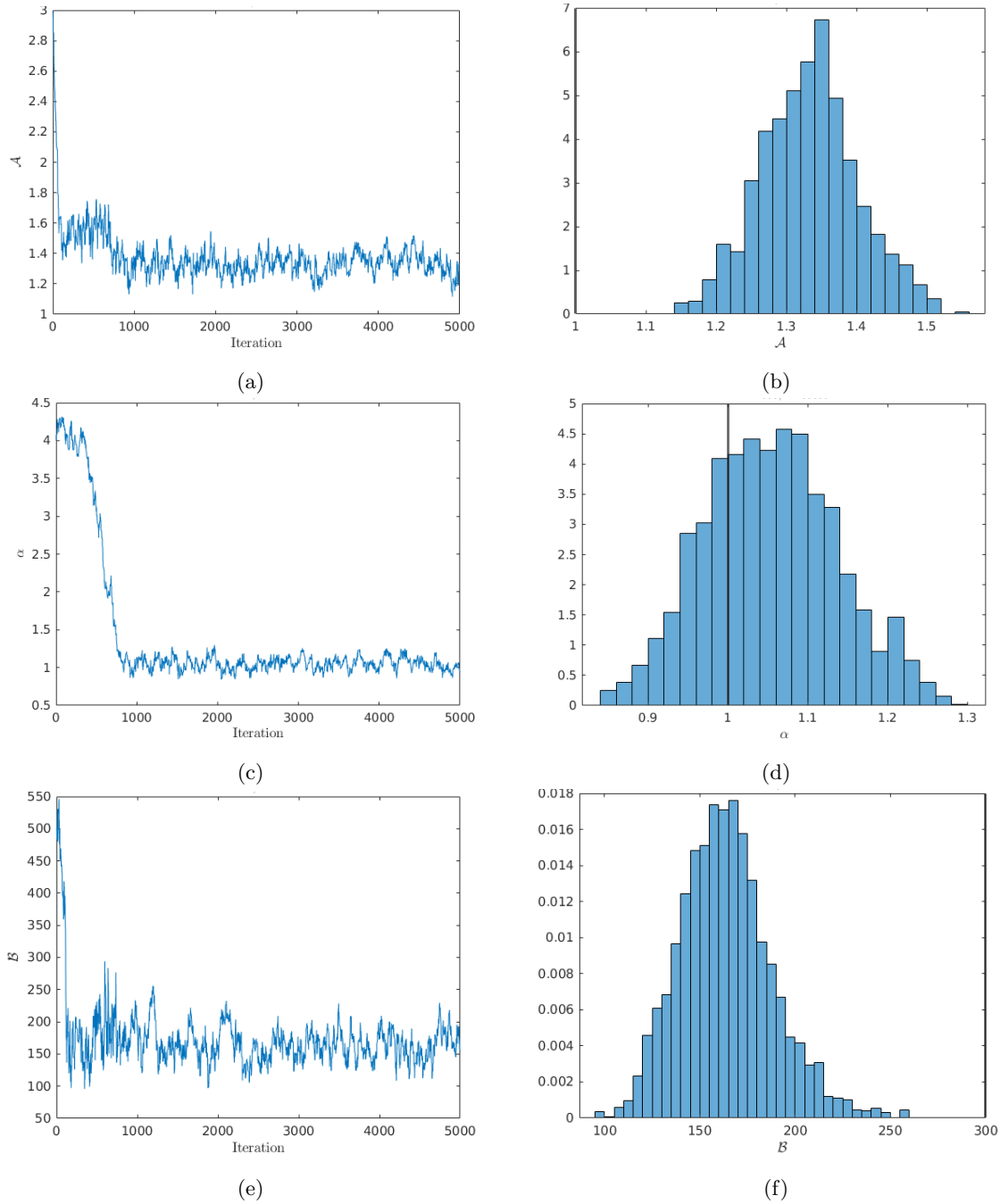


Figure C.2: (Simulation 2) Markov chains and distributions of A , α and B for one simulation with Algorithm 7. The true parameter values are $A = 1$, $\alpha = 1$ and $B = 300$. (a) Markov chain for A . (b) Distribution of A after burn-in. (c) Markov chain for α . (d) Distribution of α after burn-in. (e) Markov chain for B . (f) Distribution of B after burn-in.

