Written material for the course held in Spring 2012
Version 1.3 (March 2, 2012)

BAYESIAN ESTIMATION OF TIME-VARYING SYSTEMS:

Discrete-Time Systems

Simo Särkkä

Contents

Chapter 1

Introduction

1.1 Why The Bayesian Approach?

The mathematical treatment of the models and algorithms in this document is Bayesian, which means that all the results are treated as being approximations to certain probability distributions or their parameters. Probability distributions are used to represent both the uncertainties in the models and for modeling the physical randomness. The theory of non-linear optimal filtering is formulated in terms of Bayesian inference, and both the classical and recent filtering algorithms are derived using the same Bayesian notation and formalism.

The selection of the Bayesian approach is more a practical engineering than a philosophical decision. It simply is easier to develop a consistent, practically applicable theory of recursive inference under the Bayesian philosophy than under, for example, the least squares or the maximum likelihood philosophy. Another useful consequence of selecting the Bayesian approach is that least squares, maximum likelihood and many other philosophically different results can be obtained as special cases or re-interpretations of the Bayesian results. Of course, quite often the same thing applies also the other way around.

Modeling uncertainty as randomness is a very “engineering” way of modeling the world. It is exactly the approach also chosen in statistical physics as well as in financial analysis. The Bayesian approach to optimal filtering is far from new (see, e.g., Ho and Lee, 1964; Lee, 1964; Jazwinski, 1966; Stratonovich, 1968; Jazwinski, 1970), because the theory already existed at the same time the seminal article of Kalman (1960b) was published. The Kalman filter was first derived from the least squares point of view, but non-linear filtering theory has been Bayesian from the beginning (see, e.g., Jazwinski, 1970).

One should not take the Bayesian way of modeling unknown parameters as random variables too literally. It does not imply that one believes that there really is something random in the parameters — it is just a convenient way of representing uncertainty using the same formalism that is used for representing randomness. Random or stochastic processes appearing in the mathematical models are not necessarily really random in a physical sense, instead, the randomness is just a mathematical device for taking into account the uncertainty in a dynamic phenomenon.

But it does not matter if the randomness is interpreted as physical randomness or as a representation of uncertainty, as long as the randomness based models succeed in modeling the real world. In the above engineering philosophy the controversy between so called “frequentists” and “Bayesians” is as futile as the unnecessary controversy about interpretations of quantum mechanics, that is, whether, for example, the Copenhagen interpretation or many worlds interpretation is the correct one. The philosophical interpretation does not matter as long as we get meaningful predictions from the theory.

1.2 What is Optimal Filtering?

Optimal filtering refers to the methodology that can be used for estimating the state of a time-varying system which is indirectly observed through noisy measurements. The *state* of the system refers to the collection of dynamic variables such as position, velocity, orientation, and angular velocity, which define the physical state of the system. The *noise* in the measurements means that the measurements are uncertain in the sense that even if we knew the true system state the measurements would not be deterministic functions of the state, but would have a distribution of possible values. The time evolution of the state is modeled as a dynamic system which is perturbed by a certain *process noise*. This noise is used for modeling the uncertainties in the system dynamics. In most cases the system is not truly stochastic, but the stochasticity is only used for representing the model uncertainties.

1.2.1 Applications of Optimal Filtering

Phenomena which can be modeled as time varying systems of the above type are very common in engineering applications. These kind of models can be found, for example, in navigation, aerospace engineering, space engineering, remote surveillance, telecommunications, physics, audio signal processing, control engineering, finance and several other fields. Examples of such applications are the following:

- *Global positioning system (GPS)* (Kaplan, 1996) is a widely used satellite navigation system, where the GPS receiver unit measures arrival times of signals from several GPS satellites and computes its position based on these measurements. The GPS receiver typically uses an extended Kalman filter or some other optimal filtering algorithm for computing the position and velocity such that the measurements and the assumed dynamics (laws of physics) are taken into account. Also the ephemeris information, which is the satellite reference information transmitted from the satellites to the GPS receivers, is typically generated using optimal filters.

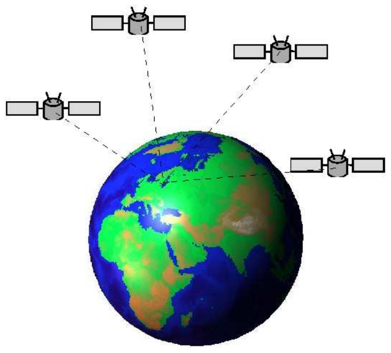


Figure 1.1: *In GPS system, the measurements are time delays of satellite signals and the optimal filter (e.g., EKF) computes the position and the accurate time.*

- *Target tracking* (Bar-Shalom et al., 2001; Crassidis and Junkins, 2004) refers to the methodology where a set of sensors such as active or passive radars, radio frequency sensors, acoustic arrays, infrared sensors and other types of sensors are used for determining the position and velocity of a remote target. When this tracking is done continuously, the dynamics of the target and measurements from the different sensors are most naturally combined using an optimal filter. The target in this (single) target tracking case can be, for example, a robot, a satellite, a car or an airplane.

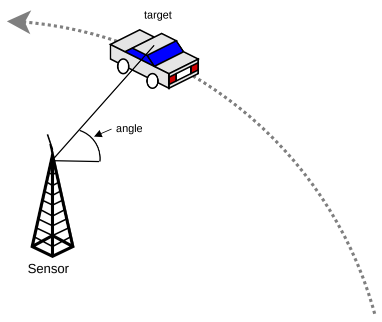


Figure 1.2: *In target tracking, a sensor generates measurements (e.g., angle measurements) of the target, and the purpose is to determine the target trajectory.*

- *Multiple target tracking* (Bar-Shalom and Li, 1995; Blackman and Popoli, 1999; Stone et al., 1999; Särkkä et al., 2007b) systems are used for remote surveillance in the cases where there are multiple targets moving at the same time in the same geographical area. This raises the concept of data association (which measurement was from which target?) and the problem of estimating the number of targets. Multiple target tracking systems are typically used in remote surveillance for military purposes, but possible civil applications are, for example, monitoring of car tunnels, automatic alarm systems and people tracking in buildings.

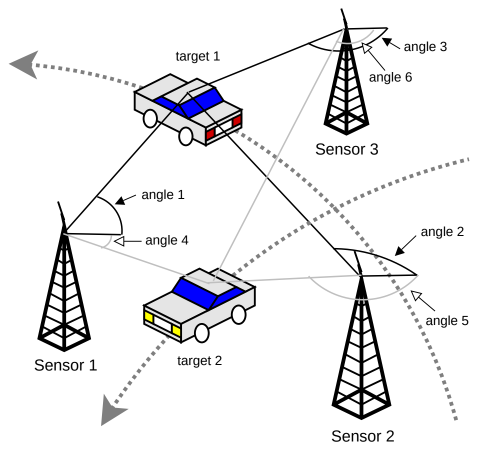


Figure 1.3: *In multiple target tracking the data association problem has to be solved, because it is impossible to know without any additional information which target produced which measurement.*

- *Inertial navigation* (Titterton and Weston, 1997; Grewal et al., 2001) uses inertial sensors such as accelerometers and gyroscopes for computing the position and velocity of a device such as a car, an airplane or a missile. When the inaccuracies in sensor measurements are taken into account the natural way of computing the estimates is by using an optimal filter. Also in sensor calibration, which is typically done in time varying environment optimal filters are often applied.

- *Integrated inertial navigation* (Grewal et al., 2001; Bar-Shalom et al., 2001) combines the good sides of unbiased but inaccurate sensors, such as altimeters and landmark trackers, and biased but locally accurate inertial sensors. Combining of these different sources of information is most naturally performed using an optimal filter such as the extended Kalman filter. This kind of approach was used, for example, in the guidance system of the Apollo 11 lunar module (Eagle), which landed on the moon in 1969.

- *GPS/INS navigation* (Grewal et al., 2001; Bar-Shalom et al., 2001) is a form of integrated inertial navigation where the inertial sensors are combined with a GPS receiver unit. In GPS/INS navigation system the short term fluctuations of the GPS can be compensated with the inertial sensors and the inertial sensor biases can be compensated with the GPS receiver. An additional advantage of this approach is that it is possible to temporarily switch to pure inertial navigation when the GPS receiver is unable to compute its position (i.e., has no fix) for some reason. This happens, for example, indoors, in tunnels and in other cases when there is no direct line-of-sight between the GPS receiver and the satellites.

- *Brain imaging* methods such as EEG, MEG and DOT are based on reconstruction of the source or diffusion field from noisy sensor data by using minimum norm estimates (MNE) and its variants (Hauk, 2004; Tarantola, 2004; Kaipio and Somersalo, 2005). The minimum norm solution can also be interpreted in Bayesian sense as a problem of estimating the field with certain prior structure from Gaussian observations. With that interpretation the estimation problem becomes equivalent to the familiar statistical inversion or generalized Gaussian process regression problem (Kaipio and Somersalo, 2005; Särkkä, 2011). Including dynamical priors then leads to a linear or non-linear spatio-temporal estimation problem, which can be solved with Kalman filters and smoothers (cf. Hiltunen et al., 2011; Särkkä et al., 2012b). The same can be done in inversion based approaches to functional Magnetic Resonance Imaging (fMRI) such as Inverse Imaging (InI) (Lin et al., 2006).

- *Spread of infectious diseases* (Anderson and May, 1991) can often be modeled as differential equations for the number of susceptible, infected and recovered/dead individuals. When uncertainties are introduced into the dynamic equations, and when the measurements are not perfect, the estimation of the spread of the disease can be formulated as an optimal filtering problem.

- *Biological processes* (Murray, 1993) such as population growth, predator-prey models and several other dynamic processes in biology can also be modeled as (stochastic) differential equations. The estimation of the states of these processes from inaccurate measurements can be formulated as an optimal filtering problem.

- *Telecommunications* is also a field where optimal filters are traditionally used. For example, optimal receivers, signal detectors and phase locked loops can be interpreted to contain optimal filters (Van Trees, 1968, 1971) as components. Also the celebrated Viterbi algorithm (Viterbi, 1967) can be interpreted as a combination of optimal filtering and optimal smoothing of the underlying hidden Markov model.

- *Audio signal processing* applications such as audio restoration (Godsill and Rayner, 1998) and audio signal enhancement (Fong et al., 2002) often use TVAR (time varying autoregressive) models as the underlying audio signal models. These kind of models can be efficiently estimated using optimal filters and smoothers.

- *Stochastic optimal control* (Maybeck, 1982b; Stengel, 1994) considers control of time varying stochastic systems. Stochastic controllers can typically be found in, for example, airplanes, cars and rockets. Optimal, in addition to the statistical optimality, means that control signal is constructed to minimize a performance cost, such as expected time to reach a predefined state, the amount of fuel consumed or average distance from a desired position trajectory. Optimal filters are typically used for estimating the states of the stochastic system and a deterministic optimal controller is constructed independently from the filter such that it uses the estimate of the filter as the known state. In theory, the optimal controller and optimal filter are not completely decoupled and the problem of constructing optimal stochastic controllers is far more challenging than constructing optimal filters and (deterministic) optimal controllers separately.

- *Learning systems* or adaptive systems can often be mathematically formulated in terms of optimal filters. The theory of stochastic differential equations has close relationship with Bayesian non-parametric modeling, machine learning and neural network modeling (MacKay, 1998; Bishop, 1995). Methods similar to the data association methods in multiple target tracking are also applicable to on-line adaptive classification (Andrieu et al., 2002). The connection between Gaussian process regression and optimal filtering has also been recently discussed in Särkkä et al. (2007a), Hartikainen and Särkkä (2010) and Särkkä and Hartikainen (2012).

- *Physical systems* which are time varying and measured through nonideal sensors can sometimes be formulated as stochastic state space models, and the time evolution of the system can be estimated using optimal filters (Kaipio and Somersalo, 2005). In Vauhkonen (1997) and more recently, for example, in Pikkarainen (2005) optimal filtering is applied to the Electrical Impedance Tomography (EIT) problem in a time varying setting.

1.2.2 Origins of Bayesian Optimal Filtering

The roots of Bayesian analysis of time dependent behavior are in the field of optimal linear filtering. The idea of constructing mathematically optimal recursive estimators was first presented for linear systems due to their mathematical simplicity and the most natural optimality criterion in both mathematical and modeling point of view was the least squares optimality. For linear systems the optimal Bayesian solution (with MMSE utility) coincides with the least squares solution, that is, the optimal least squares solution is exactly the posterior mean.

The history of optimal filtering starts from the *Wiener filter* (Wiener, 1950), which is a frequency domain solution to the problem of least squares optimal filtering of stationary Gaussian signals. The Wiener filter is still important in communication applications (Proakis, 2001), digital signal processing (Hayes, 1996) and image processing (Gonzalez and Woods, 2008). The disadvantages of the Wiener filter are that it can only be applied to stationary signals and that the construction of a Wiener filter is often mathematically demanding and these mathematics cannot be avoided (i.e., made transparent). Due to the demanding mathematics the Wiener filter can only be applied to simple low dimensional filtering problems.

The success of optimal linear filtering in engineering applications is mostly due to the seminal article of Kalman (1960b), which describes the recursive solution to the optimal discrete-time (sampled) linear filtering problem. The reason for the success is that the *Kalman filter* can be understood and applied with very much lighter mathematical machinery than the Wiener filter. Also, despite its mathematical simplicity, the Kalman filter (or actually the Kalman-Bucy filter; Kalman and Bucy, 1961) contains the Wiener filter as its limiting special case.

In the early stages of its history, the Kalman filter was soon discovered to belong to the class of Bayesian estimators (Ho and Lee, 1964; Lee, 1964; Jazwinski, 1966, 1970). An interesting historical detail is that while Kalman and Bucy were formulating the linear theory in the United States, Stratonovich was doing the pioneering work on the probabilistic (Bayesian) approach in Russia (Stratonovich, 1968; Jazwinski, 1970).

As discussed in the book of West and Harrison (1997), in the sixties, Kalman filter like recursive estimators were also used in the Bayesian community and it is not clear whether the theory of Kalman filtering or the theory of *dynamic linear models* (DLM) came first. Although these theories were originally derived from slightly different starting points, they are equivalent. Because of Kalman filter's useful connection to the theory and history of stochastic optimal control, this document approaches the Bayesian filtering problem from the Kalman filtering point of view.

Although the original derivation of the *Kalman filter* was based on the least squares approach, the same equations can be derived from the pure probabilistic Bayesian analysis. The Bayesian analysis of Kalman filtering is well covered in the classical book of Jazwinski (1970) and more recently in the book of Bar-Shalom et al. (2001). Kalman filtering, mostly because of its least squares interpretation, has widely been used in stochastic optimal control. A practical reason for this is that the inventor of the Kalman filter, Rudolph E. Kalman, has also made several contributions (Kalman, 1960a) to the theory of *linear quadratic Gaussian* (LQG) regulators, which are fundamental tools of stochastic optimal control (Stengel, 1994; Maybeck, 1982b).

1.2.3 Optimal Filtering and Smoothing as Bayesian Inference

Optimal Bayesian filtering (see, e.g. Jazwinski, 1970; Bar-Shalom et al., 2001; Doucet et al., 2001; Ristic et al., 2004) considers statistical inversion problems, where the unknown quantity is a vector valued time series $(\mathbf{x}_1, \mathbf{x}_2, \ldots)$ which is observed through noisy measurements $(\mathbf{y}_1, \mathbf{y}_2, \ldots)$ as illustrated in the Figure 1.4. An example of this kind of time series is shown in Figure 1.5. The process shown is actually a discrete-time noisy resonator with a known angular velocity. The state

$\mathbf{x}_k = (x_k \; \dot{x}_k)^T$ is two dimensional and consists of the position of the resonator x_k and its time derivative $\dot{x}_k$. The measurements y_k are scalar observations of the resonator position (signal) and they are corrupted by measurement noise.

$$\begin{tikzcd}
\text{observed:} & \mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3 & \mathbf{y}_4 & \\
\text{hidden:} & \mathbf{x}_1 \arrow[u] \arrow[r] & \mathbf{x}_2 \arrow[u] \arrow[r] & \mathbf{x}_3 \arrow[u] \arrow[r] & \mathbf{x}_4 \arrow[u] \arrow[r] & \cdots
\end{tikzcd}$$

Figure 1.4: In discrete-time filtering a sequence of hidden states $\mathbf{x}_k$ is indirectly observed through noisy measurements $\mathbf{y}_k$.

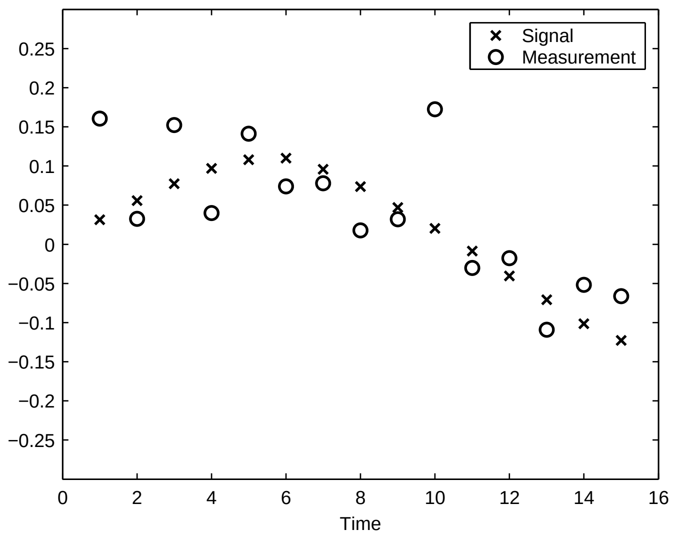


Figure 1.5: An example of time series, which models a discrete-time resonator. The actual resonator state (signal) is hidden and only observed through the noisy measurements.

The purpose of the *statistical inversion* at hand is to estimate the hidden states $\{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ given the observed measurements $\{\mathbf{y}_1, \ldots, \mathbf{y}_T\}$, which means that in the Bayesian sense (Bernardo and Smith, 1994; Gelman et al., 1995) all we have to do is to compute the joint posterior distribution of all the states given all the measurements. This can be done by straightforward application of Bayes' rule:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_T \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_T) = \frac{p(\mathbf{y}_1, \ldots, \mathbf{y}_T \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_T)\, p(\mathbf{x}_1, \ldots, \mathbf{x}_T)}{p(\mathbf{y}_1, \ldots, \mathbf{y}_T)}, \quad (1.1)$$

where

- $p(\mathbf{x}_1, \ldots, \mathbf{x}_T)$, is the prior distribution defined by the dynamic model,

- $p(\mathbf{y}_1, \ldots, \mathbf{y}_T \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_T)$ is the likelihood model for the measurements,
- $p(\mathbf{y}_1, \ldots, \mathbf{y}_T)$ is the normalization constant defined as

$$p(\mathbf{y}_1, \ldots, \mathbf{y}_T) = \int p(\mathbf{y}_1, \ldots, \mathbf{y}_T \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_T)\, p(\mathbf{x}_1, \ldots, \mathbf{x}_T)\, d(\mathbf{x}_1, \ldots, \mathbf{x}_T). \tag{1.2}$$

Unfortunately, this full posterior formulation has the serious disadvantage that each time we obtain a new measurement, the full posterior distribution would have to be recomputed. This is particularly a problem in dynamic estimation (which is exactly the problem we are solving here!), where measurements are typically obtained one at a time and we would want to compute the best possible estimate after each measurement. When the number of time steps increases, the dimensionality of the full posterior distribution also increases, which means that the computational complexity of a single time step increases. Thus eventually the computations will become intractable, no matter how much computational power is available. Without additional information or restrictive approximations, there is no way of getting over this problem in the full posterior computation.

However, the above problem only arises when we want to compute the *full* posterior distribution of the states at each time step. If we are willing to relax this a bit and be satisfied with selected marginal distributions of the states, the computations become order of magnitude lighter. In order to achieve this, we also need to restrict the class of dynamic models to probabilistic Markov sequences, which is not as restrictive as it may first seem. The model for the states and measurements will be assumed to be of the following type:

- **Initial distribution** specifies the *prior distribution* $p(\mathbf{x}_0)$ of the hidden state $\mathbf{x}_0$ at initial time step $k = 0$.
- **Dynamic model** describes the system dynamics and its uncertainties as a *Markov sequence*, defined in terms of the transition distribution $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$.
- **Measurement model** describes how the measurement $\mathbf{y}_k$ depends on the current state $\mathbf{x}_k$. This dependence is modeled by specifying the distribution of the measurement given the state, $p(\mathbf{y}_k \,|\, \mathbf{x}_k)$.

Because computing the full joint distribution of the states at all time steps is computationally very inefficient and unnecessary in real-time applications, in *optimal (Bayesian) filtering* the following marginal distributions are considered instead:

- *Filtering distributions* are the marginal distributions of *the current state* $\mathbf{x}_k$ given *the current and previous measurements* $\{\mathbf{y}_1, \ldots, \mathbf{y}_k\}$:

$$p(\mathbf{x}_k \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_k), \qquad k = 1, \ldots, T. \tag{1.3}$$

- *Prediction distributions* are the marginal distributions of the future state, n steps after the current time step:

$$p(\mathbf{x}_{k+n} \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_k), \qquad k = 1, \ldots, T, \quad n = 1, 2, \ldots, \tag{1.4}$$

- *Smoothing distributions* are the marginal distributions of the state $\mathbf{x}_k$ given a certain interval $\{\mathbf{y}_1, \ldots, \mathbf{y}_T\}$ of measurements with $T > k$:

$$p(\mathbf{x}_k \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_T), \qquad k = 1, \ldots, T. \tag{1.5}$$

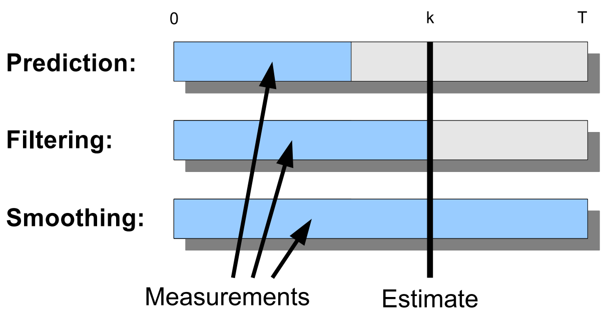


Figure 1.6: State estimation problems can be divided into optimal prediction, filtering and smoothing depending on the time span of measurements available with respect to the estimated state's time.

1.2.4 Algorithms for Optimal Filtering and Smoothing

There exists a few classes of filtering and smoothing problems which have closed form solutions:

- *Kalman filter* (KF) is a closed form solution to the discrete linear filtering problem. Due to linear Gaussian model assumptions the posterior distribution is exactly Gaussian and no numerical approximations are needed.

- *Rauch-Tung-Striebel smoother* (RTSS) is the corresponding closed form smoother to linear Gaussian state space models.

- *Grid filters and smoothers* are solutions to Markov models with finite state spaces.

But because the Bayesian optimal filtering and smoothing equations are generally computationally intractable, many kinds of numerical approximation methods have been developed, for example:

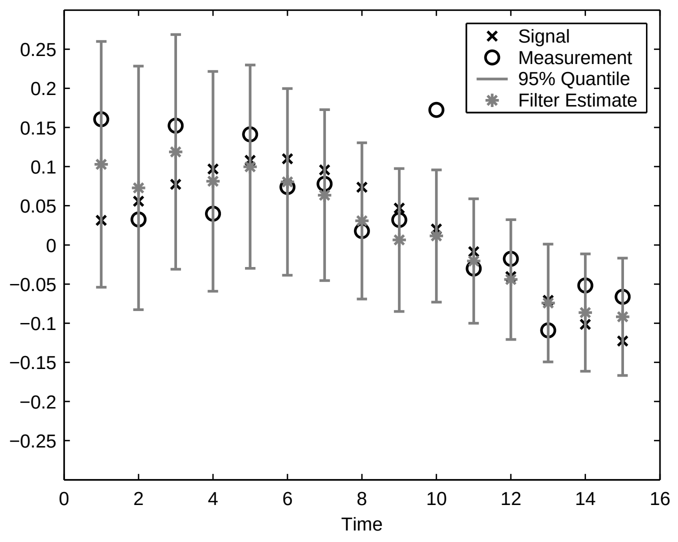


Figure 1.7: The result of computing the filtering distributions for the discrete-time resonator model. The *estimates* are the means of the filtering distributions and the quantiles are the 95% quantiles of the filtering distributions.

- *Extended Kalman filter* (EKF) approximates the non-linear and non-Gaussian measurement and dynamic models by linearization, that is, by forming a Taylor series expansion at the nominal (or Maximum a Posteriori, MAP) solution. This results in a Gaussian approximation to the filtering distribution.
- *Extended Rauch-Tung-Striebel smoother* (ERTSS) is the approximate non-linear smoothing algorithm corresponding to EKF.
- *Unscented Kalman filter* (UKF) approximates the propagation of densities through the non-linearities of measurement and noise processes by the *unscented transform*. This also results in a Gaussian approximation.
- *Unscented Rauch-Tung-Striebel smoother* (URTSS) is the approximate non-linear smoothing algorithm corresponding to UKF.
- *Sequential Monte Carlo methods* or *particle filters and smoothers* represent the posterior distribution as a weighted set of Monte Carlo samples.
- *Unscented particle filter* (UPF) and *local linearization* based methods use UKFs and EKFs, respectively, for approximating the importance distributions in sequential importance sampling.
- *Rao-Blackwellized particle filters and smoothers* use closed form integration (e.g., Kalman filters and RTS smoothers) for some of the state variables and Monte Carlo integration for others.

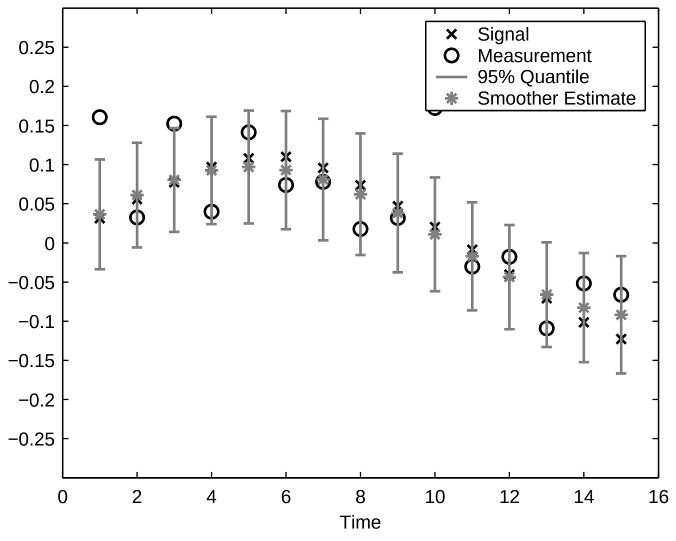


Figure 1.8: The result of computing the smoothing distributions for the discrete-time resonator model. The *estimates* are the means of the smoothing distributions and the quantiles are the 95% quantiles of the smoothing distributions. The smoothing distributions are actually the marginal distributions of the full state posterior distribution.

- *Interacting multiple models* (IMM), and other *multiple model* methods approximate the posterior distributions with a mixture of Gaussian distributions.
- *Grid based methods* approximate the distribution as a discrete distribution on a finite grid.
- *Other methods* also exist, for example, based on series expansions, describing functions, basis function expansions, exponential family of distributions, variational Bayesian methods, batch Monte Carlo (e.g., MCMC), Galerkin approximations etc.

Chapter 2

From Bayesian Inference to Bayesian Optimal Filtering

2.1 Bayesian Inference

This section provides a brief presentation of the philosophical and mathematical foundations of Bayesian inference. The connections to classical statistical inference are also briefly discussed.

2.1.1 Philosophy of Bayesian Inference

The purpose of Bayesian inference (Bernardo and Smith, 1994; Gelman et al., 1995) is to provide a mathematical machinery that can be used for modeling systems, where the uncertainties of the system are taken into account and the decisions are made according to rational principles. The tools of this machinery are the probability distributions and the rules of probability calculus.

If we compare the so called frequentist philosophy of statistical analysis to Bayesian inference the difference is that in Bayesian inference the probability of an event does not mean the proportion of the event in an infinite number of trials, but the uncertainty of the event in a single trial. Because models in Bayesian inference are formulated in terms of probability distributions, the probability axioms and computation rules of the probability theory (see, e.g., Shiryaev, 1996) also apply in the Bayesian inference.

2.1.2 Connection to Maximum Likelihood Estimation

Consider a situation where we know the conditional distribution $p(\mathbf{y}_k \,|\, \boldsymbol{\theta})$ of conditionally independent random variables (measurements) $\mathbf{y}_1, \ldots, \mathbf{y}_n$, but the parameter $\boldsymbol{\theta} \in \mathbb{R}^d$ is unknown. The classical statistical method for estimating the parameter is the *maximum likelihood method* (Milton and Arnold, 1995), where we maximize the joint probability of the measurements, also called the likelihood function

$$L(\boldsymbol{\theta}) = \prod_k p(\mathbf{y}_k \,|\, \boldsymbol{\theta}). \tag{2.1}$$

The maximum of the likelihood function with respect to $\boldsymbol{\theta}$ gives the *maximum likelihood estimate* (ML-estimate)

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}). \tag{2.2}$$

The difference between the Bayesian inference and the maximum likelihood method is that the starting point of Bayesian inference is to formally consider the parameter $\boldsymbol{\theta}$ as a random variable. Then the posterior distribution of the parameter $\boldsymbol{\theta}$ can be computed by using the *Bayes' rule*

$$p(\boldsymbol{\theta} \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_n) = \frac{p(\mathbf{y}_1, \ldots, \mathbf{y}_n \,|\, \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathbf{y}_1, \ldots, \mathbf{y}_n)}, \tag{2.3}$$

where $p(\boldsymbol{\theta})$ is the prior distribution, which models the prior beliefs of the parameter before we have seen any data and $p(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ is a normalization term, which is independent of the parameter $\boldsymbol{\theta}$. Often this normalization constant is left out and if the measurements $\mathbf{y}_1, \ldots, \mathbf{y}_n$ are conditionally independent given $\boldsymbol{\theta}$, the posterior distribution of the parameter can be written as

$$p(\boldsymbol{\theta} \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_n) \propto p(\boldsymbol{\theta}) \prod_k p(\mathbf{y}_k \,|\, \boldsymbol{\theta}). \tag{2.4}$$

Because we are dealing with a distribution, we might now choose the most probable value of the random variable (MAP-estimate), which is given by the maximum of the posterior distribution. However, an optimal estimate in mean squared sense is the posterior mean of the parameter (MMSE-estimate). There are an infinite number of other ways of choosing the point estimate from the distribution and the best way depends on the assumed loss function (or utility function). The ML-estimate can be considered as a MAP-estimate with uniform prior on the parameter $\boldsymbol{\theta}$.

One can also interpret Bayesian inference as a convenient method for including regularization terms into maximum likelihood estimation. The basic ML-framework does not have a self-consistent method for including regularization terms or prior information into statistical models. However, this regularization interpretation of Bayesian inference is not entirely right, because Bayesian inference is much more than this.

2.1.3 The Building Blocks of Bayesian Models

The basic blocks of a Bayesian model are the *prior model* containing the preliminary information on the parameter and the *measurement model* determining the stochastic mapping from the parameter to the measurements. Using the combination rules, namely Bayes' rule, it is possible to infer an estimate of the parameters from the measurements. The probability distribution of the parameters, conditional on the observed measurements is called the *posterior distribution* and it is the distribution representing the state of knowledge about the parameters when all the information in the observed measurements and the model is used. The *predictive posterior distribution* is the distribution of new (not yet observed) measurements when all the information in the observed measurements and the model is used.

- **Prior model**
  The prior information consists of subjective experience based beliefs about the possible and impossible parameter values and their relative likelihoods before anything has been observed. The prior distribution is a mathematical representation of this information:

  $$p(\boldsymbol{\theta}) = \text{Information on parameter } \boldsymbol{\theta} \text{ before seeing any observations.} \tag{2.5}$$

  The lack of prior information can be expressed by using a non-informative prior. The non-informative prior distribution can be selected in various different ways (Gelman et al., 1995).

- **Measurement model**
  Between the true parameters and the measurements there often is a causal, but inaccurate or noisy relationship. This relationship is mathematically modeled using the measurement model:

  $$p(\mathbf{y} \,|\, \boldsymbol{\theta}) = \text{Distribution of observation } \mathbf{y} \text{ given the parameters } \boldsymbol{\theta}. \tag{2.6}$$

- **Posterior distribution**
  Posterior distribution is the conditional distribution of the parameters, and it represents the information we have after the measurement $\mathbf{y}$ has been obtained. It can be computed by using the Bayes' rule:

  $$p(\boldsymbol{\theta} \,|\, \mathbf{y}) = \frac{p(\mathbf{y} \,|\, \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathbf{y})} \propto p(\mathbf{y} \,|\, \boldsymbol{\theta})\, p(\boldsymbol{\theta}), \tag{2.7}$$

  where the normalization constant is given as

  $$p(\mathbf{y}) = \int_{\mathbb{R}^d} p(\mathbf{y} \,|\, \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}. \tag{2.8}$$

  In the case of multiple measurements $\mathbf{y}_1, \ldots, \mathbf{y}_n$, if the measurements are conditionally independent the joint likelihood of all measurements is the product of distributions of individual measurements and the posterior distribution is

  $$p(\boldsymbol{\theta} \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_n) \propto p(\boldsymbol{\theta}) \prod_k p(\mathbf{y}_k \,|\, \boldsymbol{\theta}), \tag{2.9}$$

  where the normalization term can be computed by integrating the right hand side over $\boldsymbol{\theta}$. If the random variable is discrete the integration is replaced by summation.

- **Predictive posterior distribution**
  The predictive posterior distribution is the distribution of new measurements $\mathbf{y}_{n+1}$:

$$p(\mathbf{y}_{n+1} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n) = \int_{\mathbb{R}^d} p(\mathbf{y}_{n+1} \mid \boldsymbol{\theta}) \, p(\boldsymbol{\theta} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n) \, \mathrm{d}\boldsymbol{\theta}. \tag{2.10}$$

  After obtaining the measurements $\mathbf{y}_1, \ldots, \mathbf{y}_n$ the predictive posterior distribution can be used for computing the probability distribution for $n+1$:th measurement, which has not been observed yet.

In the case of tracking, we could imagine that the parameter is the sequence of dynamic states of a target, where the state contains the position and velocity. Or in the continuous-discrete setting the parameter would be the random function describing the trajectory of the target at a given time interval. In both cases the measurements could be, for example, noisy distance and direction measurements produced by a radar.

2.1.4 Bayesian Point Estimates

The distributions as such have no use in applications, but also in Bayesian computations finite dimensional summaries (point estimates) are needed. This selection of a point based on observed values of random variables is a statistical decision, and therefore this selection procedure is most naturally formulated in terms of *statistical decision theory* (Berger, 1985; Bernardo and Smith, 1994; Raiffa and Schlaifer, 2000).

Definition 2.1 (Loss Function)**.** *A loss function $L(\boldsymbol{\theta}, \mathbf{a})$ is a scalar valued function which determines the loss of taking the* action $\mathbf{a}$ *when the true parameter value is $\boldsymbol{\theta}$. The action (or control) is the statistical decision to be made based on the currently available information.*

Instead of loss functions it is also possible to work with utility functions $U(\boldsymbol{\theta}, \mathbf{a})$, which determine the reward from taking the action $\mathbf{a}$ with parameter values $\boldsymbol{\theta}$. Loss functions can be converted to utility functions and vice versa by defining $U(\boldsymbol{\theta}, \mathbf{a}) = -L(\boldsymbol{\theta}, \mathbf{a})$.

If the value of parameter $\boldsymbol{\theta}$ is not known, but the knowledge of the parameter can be expressed in terms of the posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n)$, then the natural choice is the action which gives the *minimum (maximum) of the expected loss (utility)* (Berger, 1985):

$$\mathrm{E}[L(\boldsymbol{\theta}, \mathbf{a}) \mid \mathbf{y}_1, \ldots, \mathbf{y}_n] = \int_{\mathbb{R}^d} L(\boldsymbol{\theta}, \mathbf{a}) \, p(\boldsymbol{\theta} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n) \, \mathrm{d}\boldsymbol{\theta}. \tag{2.11}$$

Commonly used loss functions are the following:

- *Quadratic error loss*: If the loss function is quadratic

$$L(\boldsymbol{\theta}, \mathbf{a}) = (\boldsymbol{\theta} - \mathbf{a})^T (\boldsymbol{\theta} - \mathbf{a}), \tag{2.12}$$

  then the optimal choice $\mathbf{a}_o$ is the *mean* of the posterior distribution of $\boldsymbol{\theta}$:

$$\mathbf{a}_o = \int_{\mathbb{R}^d} \boldsymbol{\theta}\, p(\boldsymbol{\theta} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n)\, \mathrm{d}\boldsymbol{\theta}. \tag{2.13}$$

  This posterior mean based estimate is often called the *minimum mean squared error (MMSE)* estimate of the parameter $\boldsymbol{\theta}$. The quadratic loss is the most commonly used loss function, because it is easy to handle mathematically and because in the case of Gaussian posterior distribution the MAP estimate and the median coincide with the posterior mean.

- *Absolute error loss*: The loss function of the form

$$L(\boldsymbol{\theta}, \mathbf{a}) = \sum_i |\theta_i - a_i|, \tag{2.14}$$

  is called an absolute error loss and in this case the optimal choice is the *median* of the distribution (i.e., medians of the marginal distributions in multidimensional case).

- *0-1 loss*: If the loss function is of the form

$$L(\boldsymbol{\theta}, \mathbf{a}) = -\delta(\mathbf{a} - \boldsymbol{\theta}), \tag{2.15}$$

  where $\delta(\cdot)$ is the Dirac's delta function, then the optimal choice is the maximum (mode) of the posterior distribution, that is, the *maximum a posterior (MAP)* estimate of the parameter. If the random variable $\mathbf{a}$ is discrete the corresponding loss function can be defined as

$$L(\boldsymbol{\theta}, \mathbf{a}) = \begin{cases} 0 & , \text{ if } \boldsymbol{\theta} = \mathbf{a} \\ 1 & , \text{ if } \boldsymbol{\theta} \neq \mathbf{a}. \end{cases} \tag{2.16}$$

2.1.5 Numerical Methods

In principle, Bayesian inference provides the equations for computing the posterior distributions and point estimates for any model once the model specification has been set up. However, the practical difficulty is that computation of the integrals involved in the equations can rarely be performed analytically and numerical methods are needed. Here we shall briefly describe numerical methods which are also applicable in higher dimensional problems: Gaussian approximations, multidimensional quadratures, Monte Carlo methods, and importance sampling.

- Very common types of approximations are *Gaussian approximations* (Gelman et al., 1995), where the posterior distribution is approximated with a Gaussian distribution

$$p(\boldsymbol{\theta} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n) \approx \mathrm{N}(\boldsymbol{\theta} \mid \mathbf{m}, \mathbf{P}). \tag{2.17}$$

  The mean $\mathbf{m}$ and covariance $\mathbf{P}$ of the Gaussian approximation can be either computed by matching the first two moments of the posterior distribution, or by using the mode of the distribution as the approximation of $\mathbf{m}$ and by approximating $\mathbf{P}$ using the curvature of the posterior at the mode.

- *Multi-dimensional quadrature or cubature integration methods* such as Gauss-Hermite quadrature can also be often used if the dimensionality of the integral is moderate. In those methods the idea is to deterministically form a representative set of sample points $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^{(i)} \mid i = 1, \ldots, N\}$ (sometimes called *sigma points*) and form the approximation of the integral as weighted average:

$$\mathrm{E}[\mathbf{g}(\boldsymbol{\theta}) \mid \mathbf{y}_1, \ldots, \mathbf{y}_n] \approx \sum_{i=1}^{N} W^{(i)} \, \mathbf{g}(\boldsymbol{\theta}^{(i)}), \tag{2.18}$$

  where the numerical values of the weights $W^{(i)}$ are determined by the algorithm. The sample points and weights can be selected, for example, to give exact answers for polynomials up to certain degree or to account for the moments up to certain degree.

- In direct *Monte Carlo methods* a set of N samples from the posterior distribution is randomly drawn

$$\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n), \qquad i = 1, \ldots, N, \tag{2.19}$$

  and expectation of any function $\mathbf{g}(\cdot)$ can be then approximated as the sample average

$$\mathrm{E}[\mathbf{g}(\boldsymbol{\theta}) \mid \mathbf{y}_1, \ldots, \mathbf{y}_n] \approx \frac{1}{N} \sum_{i} \mathbf{g}(\boldsymbol{\theta}^{(i)}). \tag{2.20}$$

  Another interpretation of this is that Monte Carlo methods form an approximation of the posterior density of the form

$$p(\boldsymbol{\theta} \mid \mathbf{y}_1, \ldots, \mathbf{y}_n) \approx \frac{1}{N} \sum_{i=1}^{N} \delta(x - x^{(i)}), \tag{2.21}$$

  where $\delta(\cdot)$ is the Dirac delta function. The convergence of Monte Carlo approximation is guaranteed by the *central limit theorem (CLT)* (see, e.g., Liu, 2001) and the error term is, at least in theory, independent of the dimensionality of $\boldsymbol{\theta}$. The rule of thumb is that the error decreases like the square root of the number of samples, regardless of the dimensions.

- Efficient methods for generating non-independent Monte Carlo samples are the *Markov chain Monte Carlo* (MCMC) methods (see, e.g., Gilks et al., 1996). In MCMC methods, a Markov chain is constructed such that it has the target distribution as its stationary distribution. By simulating the Markov chain, samples from the target distribution can be generated.

- *Importance sampling* (see, e.g., Liu, 2001) is a simple algorithm for generating *weighted* samples from the target distribution. The difference to the direct Monte Carlo sampling and to MCMC is that each of the particles contains a weight, which corrects the difference between the actual target distribution and the approximation obtained from an importance distribution $\pi(\cdot)$.

  Importance sampling estimate can be formed by drawing N samples from the *importance distribution*

$$\boldsymbol{\theta}^{(i)} \sim \pi(\boldsymbol{\theta} \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_n), \qquad i = 1, \ldots, N. \tag{2.22}$$

  The *importance weights* are then computed as

$$w^{(i)} = \frac{p(\boldsymbol{\theta}^{(i)} \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_n)}{\pi(\boldsymbol{\theta}^{(i)} \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_n)}, \tag{2.23}$$

  and the expectation of any function $\mathbf{g}(\cdot)$ can be then approximated as

$$\mathrm{E}[\mathbf{g}(\boldsymbol{\theta}) \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_n] \approx \frac{\sum_{i=1}^N w^{(i)} \, \mathbf{g}(\boldsymbol{\theta}^{(i)})}{\sum_{i=1}^N w^{(i)}}. \tag{2.24}$$

2.2 Batch and Recursive Estimation

In order to understand the meaning and applicability of optimal filtering and its relationship with recursive estimation, it is useful to go through an example where we solve a simple and familiar linear regression problem in a recursive manner. After that we shall generalize this concept to include a dynamic model in order to illustrate the differences in dynamic and batch estimation.

2.2.1 Batch Linear Regression

Consider the linear regression model

$$y_k = \theta_1 + \theta_2 \, t_k + \epsilon_k, \tag{2.25}$$

where we assume that the measurement noise is zero mean Gaussian with a given variance $\epsilon_k \sim \mathrm{N}(0, \sigma^2)$ and the prior distribution for parameters is Gaussian with known mean and covariance, $\boldsymbol{\theta} \sim \mathrm{N}(\mathbf{m}_0, \mathbf{P}_0)$. In the classical linear regression

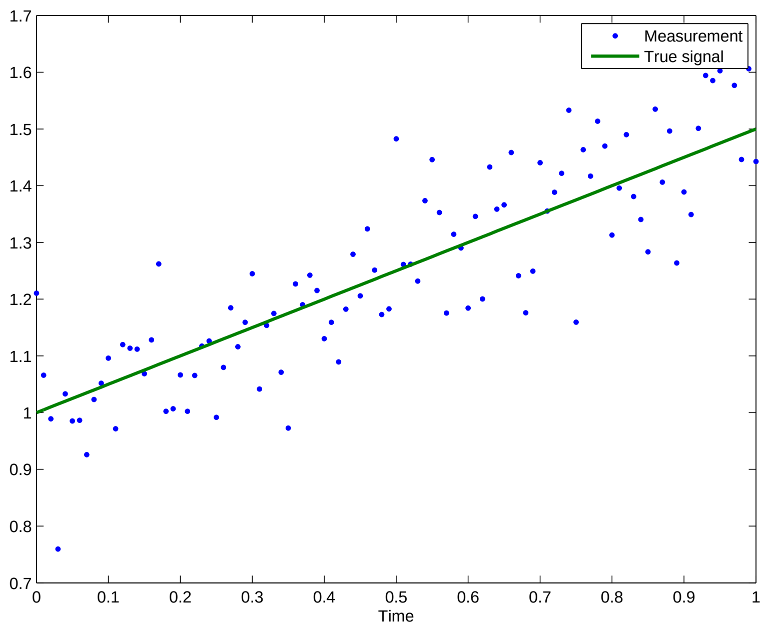


Figure 2.1: The underlying truth and the measurement data in the simple linear regression problem.

problem we want to estimate the parameters $\boldsymbol{\theta} = (\theta_1 \; \theta_2)^T$ from a set of measurement data $\mathcal{D} = \{(y_1, t_1), ..., (y_K, t_K)\}$. The measurement data and the true linear function used in simulation are illustrated in Figure 2.1.

In compact probabilistic notation the linear regression model can be written as

$$\begin{aligned} p(y_k \,|\, \boldsymbol{\theta}) &= \mathrm{N}(y_k \,|\, \mathbf{H}_k \, \boldsymbol{\theta}, \sigma^2) \\ p(\boldsymbol{\theta}) &= \mathrm{N}(\boldsymbol{\theta} \,|\, \mathbf{m}_0, \mathbf{P}_0). \end{aligned} \tag{2.26}$$

where we have introduced the matrix $\mathbf{H}_k = (1 \; t_k)$ and $\mathrm{N}(\cdot)$ denotes the Gaussian probability density function (see Appendix A.1). The likelihood of y_k is, of course, conditional on the regressors t_k also (or equivalently $\mathbf{H}_k$), but because the regressors are assumed to be known, we will not denote this dependence explicitly to simplify the notation and from now on this dependence is assumed to be understood from the context.

The *batch solution* to this linear regression problem can be obtained by straightforward application of Bayes' rule:

$$\begin{aligned} p(\boldsymbol{\theta} \,|\, y_{1:k}) &\propto p(\boldsymbol{\theta}) \prod_k p(y_k \,|\, \boldsymbol{\theta}) \\ &= \mathrm{N}(\boldsymbol{\theta} \,|\, \mathbf{m}_0, \mathbf{P}_0) \prod_k \mathrm{N}(y_k \,|\, \mathbf{H}_k \, \boldsymbol{\theta}, \sigma^2). \end{aligned}$$

Also in the posterior distribution above, we assume the conditioning on t_k and $\mathbf{H}_k$, but will not denote it explicitly. Thus the posterior distribution is denoted to be conditional on $y_{1:k} = \{y_1, \ldots, y_k\}$, and not on the data set $\mathcal{D}$ containing the regressor values t_k also. The reason for this simplification is that the simplified notation will also work in more general filtering problems, where there is no natural way of defining the associated regressor variables.

Because the prior and likelihood are Gaussian, the posterior distribution will also be Gaussian:

$$p(\boldsymbol{\theta} \,|\, y_{1:k}) = \mathrm{N}(\boldsymbol{\theta} \,|\, \mathbf{m}_K, \mathbf{P}_K). \tag{2.27}$$

The mean and covariance can be obtained by completing the quadratic form in the exponent, which gives:

$$\begin{aligned}
\mathbf{m}_K &= \left[\mathbf{P}_0^{-1} + \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H}\right]^{-1} \left[\frac{1}{\sigma^2}\mathbf{H}^T\mathbf{y} + \mathbf{P}_0^{-1}\mathbf{m}_0\right] \\
\mathbf{P}_K &= \left[\mathbf{P}_0^{-1} + \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H}\right]^{-1},
\end{aligned} \tag{2.28}$$

where $\mathbf{H}_k = (1 \; t_k)$ and

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_1 \\ \vdots \\ \mathbf{H}_K \end{pmatrix} = \begin{pmatrix} 1 & t_1 \\ \vdots & \vdots \\ 1 & t_K \end{pmatrix}, \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_K \end{pmatrix}. \tag{2.29}$$

Figure 2.2 shows the result of batch linear regression, where the posterior mean parameter values are used as the linear regression parameters.

2.2.2 Recursive Linear Regression

A *recursive solution* to the regression problem (2.26) can be obtained by assuming that we already have obtained the posterior distribution conditioned on the previous measurements $1, \ldots, k-1$:

$$p(\boldsymbol{\theta} \,|\, y_{1:k-1}) = \mathrm{N}(\boldsymbol{\theta} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1}).$$

Now assume that we have obtained a new measurement y_k and we want to compute the posterior distribution of $\boldsymbol{\theta}$ given the old measurements $y_{1:k-1}$ *and* the new measurement y_k. According to the model specification the new measurement has the likelihood

$$p(y_k \,|\, \boldsymbol{\theta}) = \mathrm{N}(y_k \,|\, \mathbf{H}_k \, \boldsymbol{\theta}, \sigma^2).$$

Using the batch version equations such that we interpret the previous posterior as the prior, we can calculate the distribution

$$\begin{aligned}
p(\boldsymbol{\theta} \,|\, y_{1:k}) &\propto p(y_k \,|\, \boldsymbol{\theta}) \, p(\boldsymbol{\theta} \,|\, y_{1:k-1}) \\
&\propto \mathrm{N}(\boldsymbol{\theta} \,|\, \mathbf{m}_k, \mathbf{P}_k),
\end{aligned} \tag{2.30}$$

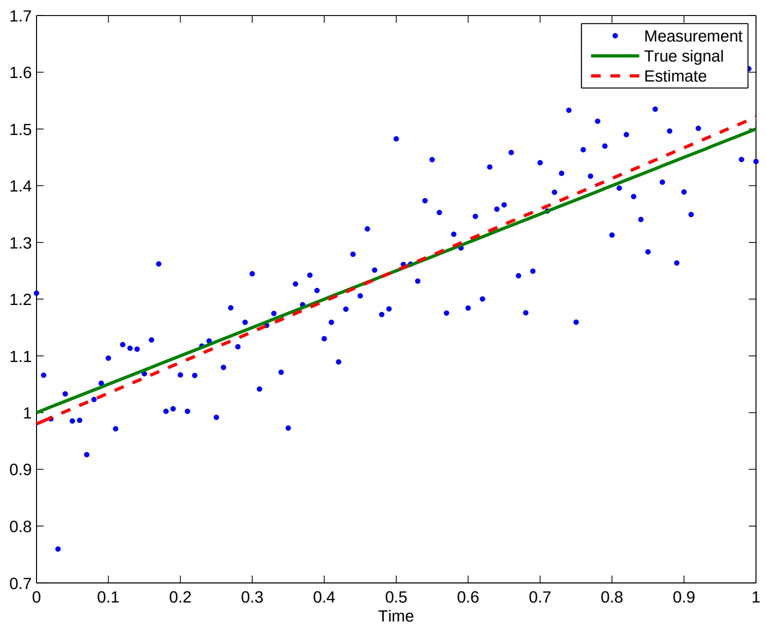


Figure 2.2: The result of simple linear regression with a slight regularization prior used for the regression parameters. For simplicity, the variance was assumed to be known.

where the Gaussian distribution parameters are

$$\begin{aligned} \mathbf{m}_k &= \left[\mathbf{P}_{k-1}^{-1} + \frac{1}{\sigma^2}\mathbf{H}_k^T\mathbf{H}_k\right]^{-1}\left[\frac{1}{\sigma^2}\mathbf{H}_k^T y_k + \mathbf{P}_{k-1}^{-1}\mathbf{m}_{k-1}\right] \\ \mathbf{P}_k &= \left[\mathbf{P}_{k-1}^{-1} + \frac{1}{\sigma^2}\mathbf{H}_k^T\mathbf{H}_k\right]^{-1}. \end{aligned} \tag{2.31}$$

By using the matrix inversion lemma , the covariance calculation can be written as

$$\mathbf{P}_k = \mathbf{P}_{k-1} - \mathbf{P}_{k-1}\mathbf{H}_k^T\left[\mathbf{H}_k\mathbf{P}_{k-1}\mathbf{H}_k^T + \sigma^2\right]^{-1}\mathbf{H}_k\mathbf{P}_{k-1}.$$

By introducing temporary variables S_k and $\mathbf{K}_k$ the calculation of mean and covariance can be written in the form

$$\begin{aligned} S_k &= \mathbf{H}_k\mathbf{P}_{k-1}\mathbf{H}_k^T + \sigma^2 \\ \mathbf{K}_k &= \mathbf{P}_{k-1}\mathbf{H}_k^T S_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_{k-1} + \mathbf{K}_k[y_k - \mathbf{H}_k\mathbf{m}_{k-1}] \\ \mathbf{P}_k &= \mathbf{P}_{k-1} - \mathbf{K}_k S_k \mathbf{K}_k^T. \end{aligned} \tag{2.32}$$

Note that $S_k = \mathbf{H}_k\mathbf{P}_{k-1}\mathbf{H}_k^T + \sigma^2$ is a scalar, because measurements are scalar and thus no matrix inversion is required.

The equations above actually are special cases of the Kalman filter update equations. Only the update part of the equations is required, because the estimated parameters are assumed to be constant, that is, there is no stochastic dynamics model for the parameters $\boldsymbol{\theta}$. Figure 2.3 illustrates the convergence of the means and variances of parameters during the recursive estimation.

2.2.3 Batch vs. Recursive Estimation

In this section we shall generalize the recursion idea used in the previous section to general probabilistic models. The underlying idea is simply that at each measurement we treat the posterior distribution of previous time step as the prior for the current time step. This way we can compute the same solution in a recursive manner that we would obtain by direct application of Bayes' rule to the whole (batch) data set.

The *batch Bayesian solution* to a statistical estimation problem can be formulated as follows:

1. Specify the likelihood model of measurements $p(\mathbf{y}_k \mid \boldsymbol{\theta})$ given the parameter $\boldsymbol{\theta}$. Typically the measurements $\mathbf{y}_k$ are assumed to be conditionally independent such that
$$p(\mathbf{y}_{1:K} \mid \boldsymbol{\theta}) = \prod_k p(\mathbf{y}_k \mid \boldsymbol{\theta}).$$
2. The prior information about the parameter $\boldsymbol{\theta}$ is encoded into the prior distribution $p(\boldsymbol{\theta})$.
3. The observed data set is $\mathcal{D} = \{(t_1, \mathbf{y}_1), \ldots, (t_K, \mathbf{y}_K)\}$, or if we drop the explicit conditioning on t_k, the data is $\mathcal{D} = \mathbf{y}_{1:K}$.
4. The batch Bayesian solution to the statistical estimation problem can be computed by applying Bayes' rule
$$p(\boldsymbol{\theta} \mid \mathbf{y}_{1:K}) = \frac{1}{Z} p(\boldsymbol{\theta}) \prod_k p(\mathbf{y}_k \mid \boldsymbol{\theta}).$$

For example, the batch solution of the above kind to the linear regression problem (2.26) was given by Equations (2.27) and (2.28).

The *recursive Bayesian solution* to the above statistical estimation problem can be formulated as follows:

1. The distribution of measurements is again modeled by the likelihood function $p(\mathbf{y}_k \mid \boldsymbol{\theta})$ and the measurements are assumed to be conditionally independent.
2. In the beginning of estimation (i.e, at step 0), all the information about the parameter $\boldsymbol{\theta}$ we have is contained in the prior distribution $p(\boldsymbol{\theta})$.

3. The measurements are assumed to be obtained one at a time, first $\mathbf{y}_1$, then $\mathbf{y}_2$ and so on. At each step we use the posterior distribution from the previous time step as the current prior distribution:

$$\begin{aligned}
p(\boldsymbol{\theta} \,|\, \mathbf{y}_1) &= \frac{1}{Z_1} p(\mathbf{y}_1 \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta}) \\
p(\boldsymbol{\theta} \,|\, \mathbf{y}_{1:2}) &= \frac{1}{Z_2} p(\mathbf{y}_2 \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta} \,|\, \mathbf{y}_1) \\
p(\boldsymbol{\theta} \,|\, \mathbf{y}_{1:3}) &= \frac{1}{Z_3} p(\mathbf{y}_3 \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta} \,|\, \mathbf{y}_{1:2}) \\
&\vdots \\
p(\boldsymbol{\theta} \,|\, \mathbf{y}_{1:K}) &= \frac{1}{Z_K} p(\mathbf{y}_K \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta} \,|\, \mathbf{y}_{1:K-1}).
\end{aligned}$$

   It is easy to show that the posterior distribution at the final step above is exactly the posterior distribution obtained by the batch solution. Also, reordering of measurements does not change the final solution.

For example, the Equations (2.30) and (2.31) give the one step update rule for the linear regression problem in Equation (2.26).

The recursive formulation of Bayesian estimation has many useful properties:

- The recursive solution can be considered as the *online learning* solution to the Bayesian learning problem. That is, the information on the parameters is updated in online manner using new pieces of information as they arrive.
- Because each step in the recursive estimation is a full Bayesian update step, *batch* Bayesian inference is a *special case of recursive* Bayesian inference.
- Due to the sequential nature of estimation we can also model the effect of time on the parameters. That is, we can model what happens to the parameter $\boldsymbol{\theta}$ between the measurements – this is actually the *basis of filtering theory*, where time behavior is modeled by assuming the parameter to be a time-dependent stochastic process $\boldsymbol{\theta}(t)$.

2.3 Towards Bayesian Filtering

Now that we are able to solve the static linear regression problem in a recursive manner, we can proceed towards Bayesian filtering by allowing the parameters to change between the measurements. By generalizing this idea, we encounter the Kalman filter, which is the workhorse of dynamic estimation.

2.3.1 Drift Model for Linear Regression

Assume that we have a similar linear regression model as in Equation (2.26), but the parameter $\boldsymbol{\theta}$ is allowed to perform *Gaussian random walk* between the measurements:

$$\begin{aligned} p(y_k \mid \boldsymbol{\theta}_k) &= \mathrm{N}(y_k \mid \mathbf{H}_k\, \boldsymbol{\theta}_k, \sigma^2) \\ p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1}) &= \mathrm{N}(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1}, \mathbf{Q}) \\ p(\boldsymbol{\theta}_0) &= \mathrm{N}(\boldsymbol{\theta}_0 \mid \mathbf{m}_0, \mathbf{P}_0), \end{aligned} \tag{2.33}$$

where $\mathbf{Q}$ is the covariance of the random walk. Now, given the distribution

$$p(\boldsymbol{\theta}_{k-1} \mid y_{1:k-1}) = \mathrm{N}(\boldsymbol{\theta}_{k-1} \mid \mathbf{m}_{k-1}, \mathbf{P}_{k-1}),$$

the joint distribution of $\boldsymbol{\theta}_k$ and $\boldsymbol{\theta}_{k-1}$ is[1]

$$p(\boldsymbol{\theta}_k, \boldsymbol{\theta}_{k-1} \mid y_{1:k-1}) = p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1})\, p(\boldsymbol{\theta}_{k-1} \mid y_{1:k-1}).$$

The distribution of $\boldsymbol{\theta}_k$ given the measurement history up to time step $k-1$ can be calculated by integrating over $\boldsymbol{\theta}_{k-1}$

$$p(\boldsymbol{\theta}_k \mid y_{1:k-1}) = \int p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1})\, p(\boldsymbol{\theta}_{k-1} \mid y_{1:k-1})\, d\boldsymbol{\theta}_{k-1}.$$

This relationship is sometimes called the *Chapman-Kolmogorov equation*. Because $p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1})$ and $p(\boldsymbol{\theta}_{k-1} \mid y_{1:k-1})$ are Gaussian, the result of the marginalization is Gaussian:

$$p(\boldsymbol{\theta}_k \mid y_{1:k-1}) = \mathrm{N}(\boldsymbol{\theta}_k \mid \mathbf{m}_k^-, \mathbf{P}_k^-),$$

where

$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{m}_{k-1} \\ \mathbf{P}_k^- &= \mathbf{P}_{k-1} + \mathbf{Q}. \end{aligned}$$

By using this as the prior distribution for the measurement likelihood $p(y_k \mid \boldsymbol{\theta}_k)$ we get the parameters of the posterior distribution

$$p(\boldsymbol{\theta}_k \mid y_{1:k}) = \mathrm{N}(\boldsymbol{\theta}_k \mid \mathbf{m}_k, \mathbf{P}_k),$$

which are given by equations (2.32), when $\mathbf{m}_{k-1}$ and $\mathbf{P}_{k-1}$ are replaced by $\mathbf{m}_k^-$ and $\mathbf{P}_k^-$:

$$\begin{aligned} S_k &= \mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \sigma^2 \\ \mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^T S_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k [y_k - \mathbf{H}_k \mathbf{m}_k^-] \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k S_k \mathbf{K}_k^T. \end{aligned} \tag{2.34}$$

This recursive computational algorithm for the time-varying linear regression weights is again a special case of the Kalman filter algorithm. Figure 2.4 shows the result of recursive estimation of a sine signal assuming a small diagonal Gaussian drift model for the parameters.

[1] Note that this formula is correct only for Markovian dynamic models, where $p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1}, y_{1:k-1}) = p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1})$.

At this point we shall change from the *regression notation* used so far into *state space model notation* , which is commonly used in Kalman filtering and related dynamic estimation literature. Because this notation easily causes confusion to people who have got used to regression notation, this point is emphasized:

- In *state space notation* $\mathbf{x}$ means the unknown state of the system, that is, the vector of *unknown parameters in the system*. It is *not* the regressor, covariate or input variable of the system.

- For example, the time-varying linear regression model with drift presented in this section can be transformed into more standard *state space model notation* by replacing the variable $\boldsymbol{\theta}_k = (\theta_{1,k}\ \theta_{2,k})^T$ with the variable $\mathbf{x}_k = (x_{1,k}\ x_{2,k})^T$:

$$\begin{aligned} p(y_k \,|\, \mathbf{x}_k) &= \mathrm{N}(y_k \,|\, \mathbf{H}_k\, \mathbf{x}_k, \sigma^2) \\ p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{Q}) \\ p(\mathbf{x}_0) &= \mathrm{N}(\mathbf{x}_0 \,|\, \mathbf{m}_0, \mathbf{P}_0). \end{aligned} \tag{2.35}$$

2.3.2 Kalman Filter for Linear Model with Drift

The linear model with drift in the previous section had the disadvantage that the covariates t_k occurred explicitly in the model specification. The problem with this is that when we get more and more measurements, the parameter t_k grows without a bound. Thus the conditioning of the problem also gets worse in time. For practical reasons it also would be desirable to have time-invariant model, that is, a model which is not dependent on the absolute time, but only on the relative positions of states and measurements in time.

The alternative state space formulation of the linear model with drift, without using explicit covariates can be done as follows. Let's denote time difference between consecutive times as $\Delta t_{k-1} = t_k - t_{k-1}$. The idea is that if the underlying phenomenon (signal, state, parameter) x_k was exactly linear, the difference between adjacent time points could be written exactly as

$$x_k - x_{k-1} = \dot{x}\, \Delta t_{k-1} \tag{2.36}$$

where $\dot{x}$ is the derivative, which is constant in the exactly linear case. The divergence from the exactly linear function can be modeled by assuming that the above equation does not hold exactly, but there is a small noise term on the right hand side. The derivative can also be assumed to perform small random walk and thus not be exactly constant. This model can be written as follows:

$$\begin{aligned} x_{1,k} &= x_{1,k-1} + \Delta t_{k-1} x_{2,k-1} + w_1 \\ x_{2,k} &= x_{2,k-1} + w_2 \\ y_k &= x_{1,k} + e, \end{aligned} \tag{2.37}$$

where the signal is the first components of the state $x_{1,k}$ and the derivative is the second $x_{2,k}$. The noises are $e \sim N(0,\sigma^2)$, $(w_1; w_2) \sim \mathrm{N}(0, \mathbf{Q})$. The model can also be written in form

$$\begin{aligned} p(y_k \,|\, \mathbf{x}_k) &= \mathrm{N}(y_k \,|\, \mathbf{H}\, \mathbf{x}_k, \sigma^2) \\ p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{A}_{k-1}\, \mathbf{x}_{k-1}, \mathbf{Q}), \end{aligned} \tag{2.38}$$

where

$$\mathbf{A}_{k-1} = \begin{pmatrix} 1 & \Delta t_{k-1} \\ 0 & 1 \end{pmatrix}, \qquad \mathbf{H} = \begin{pmatrix} 1 & 0 \end{pmatrix}.$$

With suitable $\mathbf{Q}$ this model is actually equivalent to model (2.33), but in this formulation we explicitly estimate the state of the signal (point on the regression line) instead of the linear regression parameters.

We could now explicitly derive the recursion equations in the same manner as we did in the previous sections. However, we can also use the *Kalman filter*, which is a readily derived recursive solution to generic linear Gaussian models of the form

$$\begin{aligned} p(\mathbf{y}_k \,|\, \mathbf{x}_k) &= \mathrm{N}(\mathbf{y}_k \,|\, \mathbf{H}_k\, \mathbf{x}_k, \mathbf{R}_k) \\ p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{A}_{k-1}\, \mathbf{x}_{k-1}, \mathbf{Q}_{k-1}). \end{aligned}$$

Our alternative linear regression model in Equation (2.37) can be seen to be a special case of these models. The Kalman filter equations are often expressed as prediction and update steps as follows:

1. *Prediction step:*

$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{A}_{k-1}\, \mathbf{m}_{k-1} \\ \mathbf{P}_k^- &= \mathbf{A}_{k-1}\, \mathbf{P}_{k-1}\, \mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}. \end{aligned}$$

2. *Update step:*

$$\begin{aligned} \mathbf{S}_k &= \mathbf{H}_k\, \mathbf{P}_k^-\, \mathbf{H}_k^T + \mathbf{R}_k \\ \mathbf{K}_k &= \mathbf{P}_k^-\, \mathbf{H}_k^T\, \mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, [\mathbf{y}_k - \mathbf{H}_k\, \mathbf{m}_k^-] \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T. \end{aligned}$$

The result of tracking the sine signal with Kalman filter is shown in Figure 2.5. All the mean and covariance calculation equations given in this document so far have been special cases of the above equations, including the batch solution to the scalar measurement case (which is a one-step solution). The Kalman filter recursively computes the mean and covariance of the posterior distributions of the form

$$p(\mathbf{x}_k \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_k) = \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k).$$

Note that the estimates of $\mathbf{x}_k$ derived from this distribution are non-anticipative in the sense that they are only conditional to measurements obtained before and at the time step k. However, after we have obtained measurements $\mathbf{y}_1, \ldots, \mathbf{y}_k$, we could compute estimates of $\mathbf{x}_{k-1}, \mathbf{x}_{k-2}, \ldots$, which are also conditional to the measurements after the corresponding state time steps. Because more measurements and more information is available for the estimator, these estimates can be expected to be more accurate than the non-anticipative measurements computed by the filter.

The above mentioned problem of computing estimates of the state by conditioning not only on previous measurements, but also on future measurements is called *optimal smoothing* as already mentioned in Section 1.2.3. The optimal smoothing solution to the linear Gaussian state space models is given by the *Rauch-Tung-Striebel smoother*. The full Bayesian theory of optimal smoothing as well as the related algorithms will be presented in Chapter 4.

It is also possible to predict the time behavior of the state in the future that we have not yet measured. This procedure is called *optimal prediction*. Because optimal prediction can always be done by iterating the prediction step of the optimal filter, no specialized algorithms are needed for this.

The non-linear generalizations of optimal prediction, filtering and smoothing can be obtained by replacing the Gaussian distributions and linear functions in model (2.38) with non-Gaussian and non-linear ones. The Bayesian dynamic estimation theory described in this document can be applied to generic non-linear filtering models of the following form:

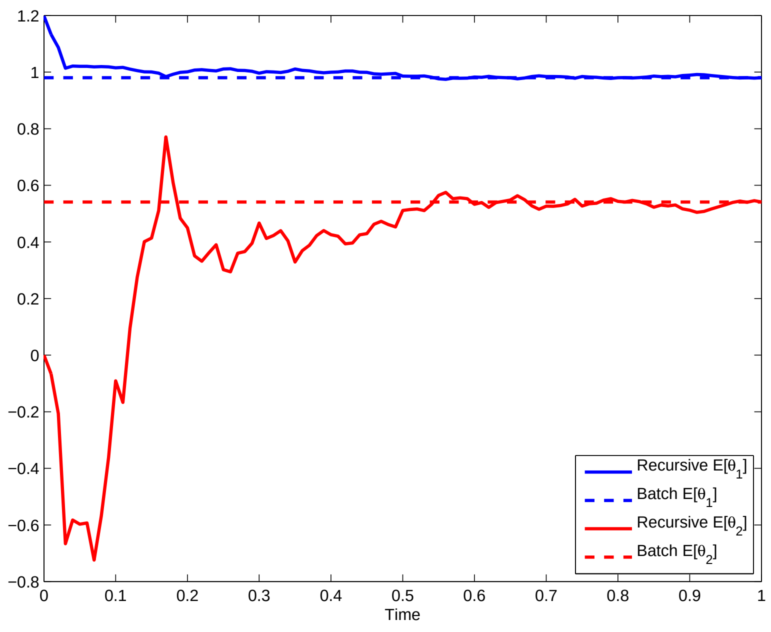
$$\begin{aligned} \text{measurement model: } & \mathbf{y}_k \sim p(\mathbf{y}_k \,|\, \mathbf{x}_k) \\ \text{state model: } & \mathbf{x}_k \sim p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}). \end{aligned}$$

To understand the generality of this model is it useful to note that if we dropped the time-dependence from the state we would get the model

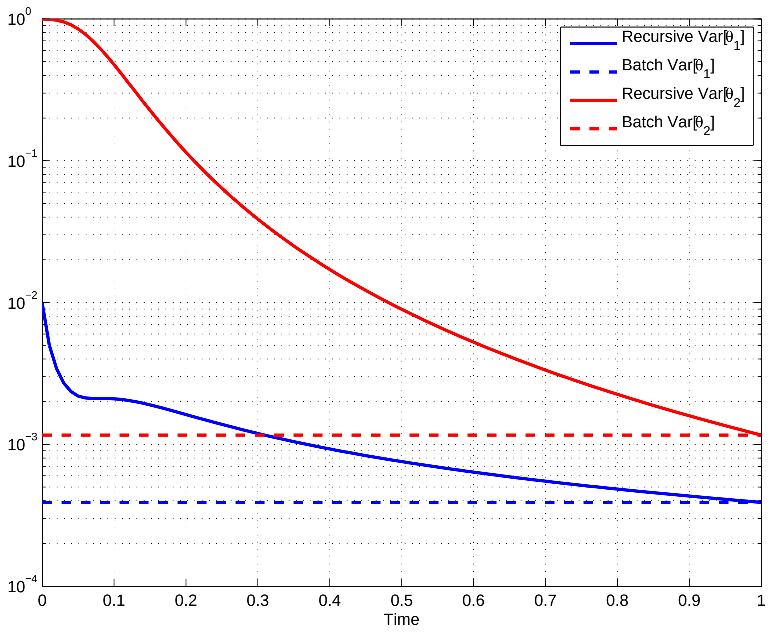
$$\begin{aligned} \text{measurement model: } & \mathbf{y}_k \sim p(\mathbf{y}_k \,|\, \mathbf{x}) \\ \text{state model: } & \mathbf{x} \sim p(\mathbf{x}). \end{aligned}$$

Because $\mathbf{x}$ denotes an arbitrary set of parameters or hyper-parameters of the system, all static Bayesian models are special cases of this model. Thus in dynamic estimation context we extend the static models by allowing a Markov model for the time-behavior of the (hyper)parameters.

The Markovianity also is less of a restriction than it sounds, because what we have is a vector valued Markov process, not a scalar one. The reader may recall from elementary calculus that differential equations of an arbitrary order can be always transformed into vector valued differential equations of the first order. In analogous manner, Markov processes of an arbitrary order can be transformed into vector valued first order Markov processes.



(a)



(b)

Figure 2.3: (a) Convergence of the recursive linear regression means. The final value is exactly the same as that was obtained with batch linear regression. Note that time has been scaled to 1 at $k = K$. (b) Convergence of the variances plotted on logarithmic scale. As can be seen, every measurement brings more information and the uncertainty decreases monotonically. The final values are the same as the variances obtained from the batch solution.

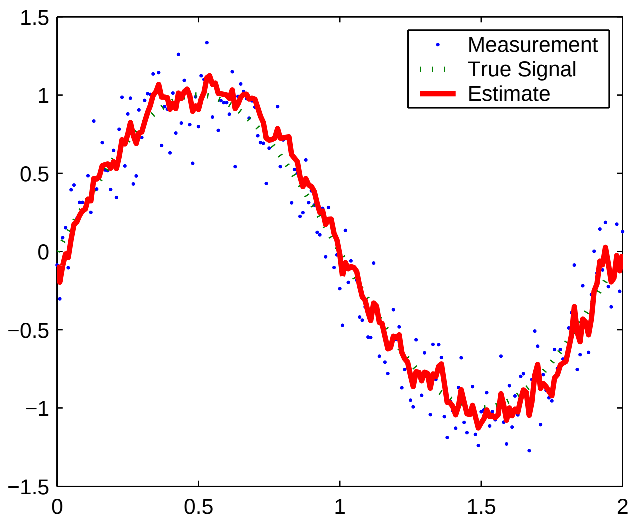


Figure 2.4: Example of tracking a sine signal with linear model with drift, where the parameters are allowed to vary according to Gaussian random walk model.

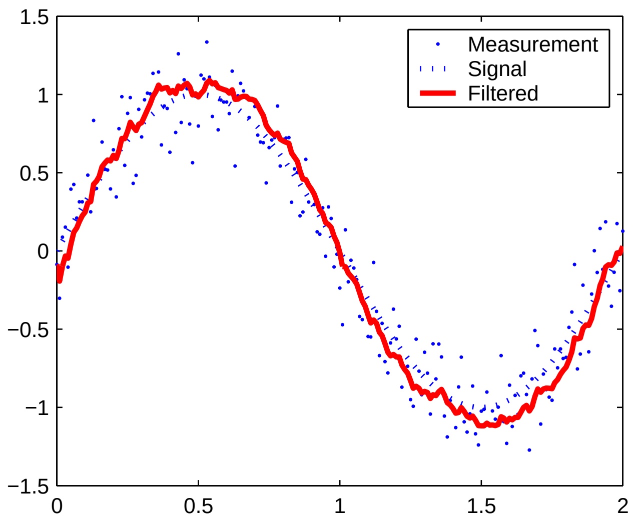


Figure 2.5: Example of tracking a sine signal with a locally linear state space model. The result differs a bit from the random walk parameter model, because of slightly different choice of process noise. It could be made equivalent if desired.

Chapter 3

Optimal Filtering

In this chapter we first present the classical formulation of discrete-time optimal filtering as recursive Bayesian inference. Then the classical Kalman filter, extended Kalman filters and statistical linearization based filters are presented in terms of the general theory. In addition to the classical algorithms the unscented Kalman filter, general Gaussian filters, Gauss-Hermite Kalman filters, Fourier-Hermite Kalman filters, and cubature Kalman filters are also presented. Sequential importance resampling based particle filtering, as well as Rao-Blackwellized particle filtering are also covered.

For more information, reader is referred to various articles and books cited in the appropriate sections. The following books also contain useful information on the subject:

- Classic books: Lee (1964); Bucy and Joseph (1968); Meditch (1969); Jazwinski (1970); Sage and Melsa (1971); Gelb (1974); Anderson and Moore (1979); Maybeck (1979, 1982a).

- More recent books on linear and non-linear Kalman filtering: Bar-Shalom et al. (2001); Grewal and Andrews (2001); Crassidis and Junkins (2004).

- Recent books with particle filters also: Ristic et al. (2004); Candy (2009); Challa et al. (2011); Crisan and Rozovskii (2011).

3.1 Formal Filtering Equations and Exact Solutions

3.1.1 Probabilistic State Space Models

Before going into the practical non-linear filtering algorithms, in the next sections the theory of probabilistic (Bayesian) filtering is presented. The Kalman filtering equations, which are the closed form solutions to the linear Gaussian discrete-time optimal filtering problem, are also derived.

Definition 3.1 (State space model)**.** A discrete-time state space model *or probabilistic non-linear filtering model consists of a sequence of conditional probability distributions:*

$$\begin{aligned} \mathbf{x}_k &\sim p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) \\ \mathbf{y}_k &\sim p(\mathbf{y}_k \,|\, \mathbf{x}_k), \end{aligned} \tag{3.1}$$

for $k = 1, 2, \ldots$, where

- $\mathbf{x}_k \in \mathbb{R}^n$ *is the* state *of the system at time step k.*
- $\mathbf{y}_k \in \mathbb{R}^m$ *is the measurement at time step k.*
- $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$ *is the* dynamic model *which describes the stochastic dynamics of the system. The dynamic model can be a probability density, a counting measure or a combination of them depending on whether the state $\mathbf{x}_k$ is continuous, discrete or hybrid.*
- $p(\mathbf{y}_k \,|\, \mathbf{x}_k)$ *is the* measurement model*, which is the distribution of measurements given the state.*

The model is assumed to be Markovian, which means that it has the following two properties:

Property 3.1 (Markov property of states)**.**

The states $\{\mathbf{x}_k : k = 0, 1, 2, \ldots\}$ form a Markov sequence (or Markov chain if the state is discrete). This Markov property means that $\mathbf{x}_k$ (and actually the whole future $\mathbf{x}_{k+1}, \mathbf{x}_{k+2}, \ldots$) given $\mathbf{x}_{k-1}$ is independent of anything that has happened before the time step $k-1$:

$$p(\mathbf{x}_k \,|\, \mathbf{x}_{1:k-1}, \mathbf{y}_{1:k-1}) = p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}). \tag{3.2}$$

Also the past is independent of the future given the present:

$$p(\mathbf{x}_{k-1} \,|\, \mathbf{x}_{k:T}, \mathbf{y}_{k:T}) = p(\mathbf{x}_{k-1} \,|\, \mathbf{x}_k). \tag{3.3}$$

Property 3.2 (Conditional independence of measurements)**.**

The current measurement $\mathbf{y}_k$ given the current state $\mathbf{x}_k$ is conditionally independent of the measurement and state histories:

$$p(\mathbf{y}_k \,|\, \mathbf{x}_{1:k}, \mathbf{y}_{1:k-1}) = p(\mathbf{y}_k \,|\, \mathbf{x}_k). \tag{3.4}$$

A simple example of a Markovian sequence is the Gaussian random walk. When this is combined with noisy measurements, we obtain the following example of a probabilistic state space model.

Example 3.1 (Gaussian random walk)**.** *A Gaussian random walk model can be written as*

$$\begin{aligned} x_k &= x_{k-1} + w_{k-1}, & w_{k-1} &\sim \mathrm{N}(0, q) \\ y_k &= x_k + e_k, & e_k &\sim \mathrm{N}(0, r), \end{aligned} \tag{3.5}$$

where x_k is the hidden state and y_k is the measurement. In terms of probability densities the model can be written as

$$\begin{aligned} p(x_k \,|\, x_{k-1}) &= \mathrm{N}(x_k \,|\, x_{k-1}, q) \\ &= \frac{1}{\sqrt{2\pi q}} \exp\left(-\frac{1}{2q} (x_k - x_{k-1})^2 \right) \\ p(y_k \,|\, x_k) &= \mathrm{N}(y_k \,|\, x_k, r) \\ &= \frac{1}{\sqrt{2\pi r}} \exp\left(-\frac{1}{2r} (y_k - x_k)^2 \right), \end{aligned} \tag{3.6}$$

which is a discrete-time state space model.

With the Markovian assumption and the filtering model (3.1), the joint prior distribution of the states $(\mathbf{x}_0, \ldots, \mathbf{x}_T)$, and the joint likelihood of the measurements $(\mathbf{y}_0, \ldots, \mathbf{y}_T)$ are, respectively

$$p(\mathbf{x}_0, \ldots, \mathbf{x}_T) = p(\mathbf{x}_0) \prod_{k=1}^{T} p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) \tag{3.7}$$

$$p(\mathbf{y}_1, \ldots, \mathbf{y}_T \,|\, \mathbf{x}_0, \ldots, \mathbf{x}_T) = \prod_{k=1}^{T} p(\mathbf{y}_k \,|\, \mathbf{x}_k). \tag{3.8}$$

In principle, for a given T we could simply compute the posterior distribution of the states by Bayes' rule:

$$\begin{aligned} p(\mathbf{x}_0, \ldots, \mathbf{x}_T \,|\, \mathbf{y}_1, \ldots, \mathbf{y}_T) &= \frac{p(\mathbf{y}_1, \ldots, \mathbf{y}_T \,|\, \mathbf{x}_0, \ldots, \mathbf{x}_T)\, p(\mathbf{x}_0, \ldots, \mathbf{x}_T)}{p(\mathbf{y}_1, \ldots, \mathbf{y}_T)} \\ &\propto p(\mathbf{y}_1, \ldots, \mathbf{y}_T \,|\, \mathbf{x}_0, \ldots, \mathbf{x}_T)\, p(\mathbf{x}_0, \ldots, \mathbf{x}_T). \end{aligned} \tag{3.9}$$

However, this kind of explicit usage of the full Bayes' rule is not feasible in real-time applications, because the amount of computations per time step increases as new observations arrive. Thus, this way we could only work with small data sets, because if the amount of data is unbounded (as in real time-sensoring applications), then at some point of time the computations would become intractable. To cope with real-time data we need to have an algorithm which does constant amount of computations per time step.

As discussed in Section 1.2.3, *filtering distributions*, *prediction distributions* and *smoothing distributions* can be computed recursively such that only constant amount of computations is done on each time step. For this reason we shall not consider the full posterior computation at all, but concentrate to the above-mentioned distributions instead. In this chapter, we mainly consider computation of the filtering and prediction distributions, and algorithms for computing the smoothing distributions will be considered in the next chapter.

3.1.2 Optimal Filtering Equations

The purpose of *optimal filtering* is to compute the *marginal posterior distribution* of the state $\mathbf{x}_k$ at each time step k given the history of the measurements up to the time step k:

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}). \tag{3.10}$$

The fundamental equations of the Bayesian filtering theory are given by the following theorem:

Theorem 3.1 (Bayesian optimal filtering equations)**.** *The recursive equations for computing the* predicted distribution $p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1})$ *and the* filtering distribution $p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k})$ *at the time step k are given by the following* Bayesian filtering equations*:*

- Initialization. *The recursion starts from the prior distribution* $p(\mathbf{x}_0)$.
- Prediction. *The predictive distribution of the state $\mathbf{x}_k$ on time step k given the dynamic model can be computed by the Chapman-Kolmogorov equation*

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) \, p(\mathbf{x}_{k-1} \,|\, \mathbf{y}_{1:k-1}) \, \mathrm{d}\mathbf{x}_{k-1}. \tag{3.11}$$

- Update. *Given the measurement $\mathbf{y}_k$ at time step k the posterior distribution of the state $\mathbf{x}_k$ can be computed by Bayes' rule*

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) = \frac{1}{Z_k} p(\mathbf{y}_k \,|\, \mathbf{x}_k) \, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}), \tag{3.12}$$

  where the normalization constant Z_k is given as

$$Z_k = \int p(\mathbf{y}_k \,|\, \mathbf{x}_k) \, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) \, \mathrm{d}\mathbf{x}_k. \tag{3.13}$$

If some of the components of the state are discrete, the corresponding integrals are replaced with summations.

Proof. The joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ given $\mathbf{y}_{1:k-1}$ can be computed as

$$\begin{aligned} p(\mathbf{x}_k, \mathbf{x}_{k-1} \,|\, \mathbf{y}_{1:k-1}) &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{y}_{1:k-1}) \, p(\mathbf{x}_{k-1} \,|\, \mathbf{y}_{1:k-1}) \\ &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) \, p(\mathbf{x}_{k-1} \,|\, \mathbf{y}_{1:k-1}), \end{aligned} \tag{3.14}$$

where the disappearance of the measurement history $\mathbf{y}_{1:k-1}$ is due to the Markov property of the sequence $\{\mathbf{x}_k, k = 1, 2, \ldots\}$. The marginal distribution of $\mathbf{x}_k$ given $\mathbf{y}_{1:k-1}$ can be obtained by integrating the distribution (3.14) over $\mathbf{x}_{k-1}$, which gives the *Chapman-Kolmogorov equation*

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) \, p(\mathbf{x}_{k-1} \,|\, \mathbf{y}_{1:k-1}) \, \mathrm{d}\mathbf{x}_{k-1}. \tag{3.15}$$

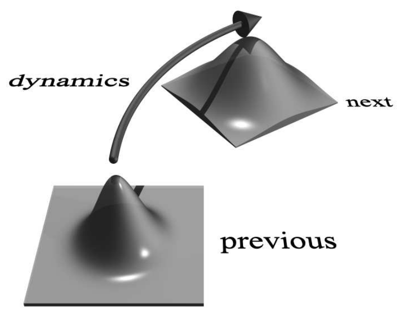


Figure 3.1: *Visualization of the prediction step: the prediction propagates the state distribution of the previous measurement step through the dynamic model such that the uncertainties (stochastic terms) in the dynamic model are taken into account.*

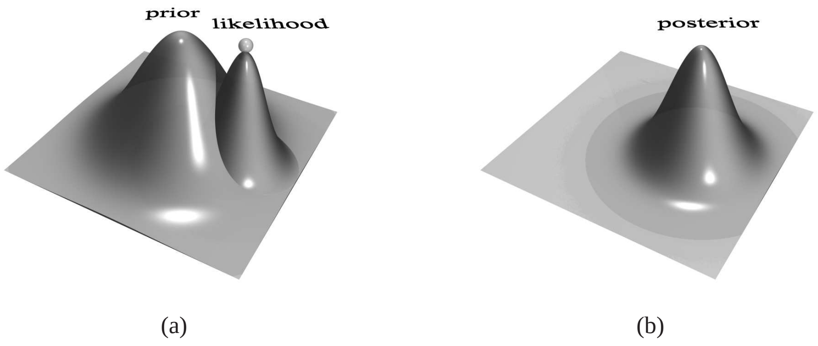


Figure 3.2: *Visualization of the update step: (a) Prior distribution from prediction and the likelihood of measurement just before the update step. (b) The posterior distribution after combining the prior and likelihood by Bayes' rule.*

If $\mathbf{x}_{k-1}$ is discrete, then the above integral is replaced with summation over $\mathbf{x}_{k-1}$. The distribution of $\mathbf{x}_k$ given $\mathbf{y}_k$ and $\mathbf{y}_{1:k-1}$, that is, given $\mathbf{y}_{1:k}$ can be computed by *Bayes' rule*

$$\begin{aligned} p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) &= \frac{1}{Z_k} p(\mathbf{y}_k \,|\, \mathbf{x}_k, \mathbf{y}_{1:k-1})\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) \\ &= \frac{1}{Z_k} p(\mathbf{y}_k \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) \end{aligned} \tag{3.16}$$

where the normalization constant is given by Equation (3.13). The disappearance of the measurement history $\mathbf{y}_{1:k-1}$ in Equation (3.16) is due to the conditional independence of $\mathbf{y}_k$ of the measurement history, given $\mathbf{x}_k$. □

3.1.3 Kalman Filter

The Kalman filter (Kalman, 1960b) is the closed form solution to the optimal filtering equations of the discrete-time filtering model, where the dynamic and measurements models are linear Gaussian:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{A}_{k-1}\,\mathbf{x}_{k-1} + \mathbf{q}_{k-1} \\ \mathbf{y}_k &= \mathbf{H}_k\,\mathbf{x}_k + \mathbf{r}_k, \end{aligned} \tag{3.17}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state, $\mathbf{y}_k \in \mathbb{R}^m$ is the measurement, $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ is the process noise, $\mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k)$ is the measurement noise and the prior distribution is Gaussian $\mathbf{x}_0 \sim \mathrm{N}(\mathbf{m}_0, \mathbf{P}_0)$. The matrix $\mathbf{A}_{k-1}$ is the transition matrix of the dynamic model and $\mathbf{H}_k$ is the measurement model matrix. In probabilistic terms the model is

$$\begin{aligned} p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}) &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{A}_{k-1}\,\mathbf{x}_{k-1}, \mathbf{Q}_{k-1}) \\ p(\mathbf{y}_k \,|\, \mathbf{x}_k) &= \mathrm{N}(\mathbf{y}_k \,|\, \mathbf{H}_k\,\mathbf{x}_k, \mathbf{R}_k). \end{aligned} \tag{3.18}$$

Algorithm 3.1 (Kalman filter)**.** *The optimal filtering equations for the linear filtering model* (3.17) *can be evaluated in closed form and the resulting distributions are Gaussian:*

$$\begin{aligned} p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-) \\ p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k) \\ p(\mathbf{y}_k \,|\, \mathbf{y}_{1:k-1}) &= \mathrm{N}(\mathbf{y}_k \,|\, \mathbf{H}_k\mathbf{m}_k^-, \mathbf{S}_k). \end{aligned} \tag{3.19}$$

The parameters of the distributions above can be computed with the following Kalman filter prediction *and* update steps*:*

- The prediction step *is*

$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{A}_{k-1}\,\mathbf{m}_{k-1} \\ \mathbf{P}_k^- &= \mathbf{A}_{k-1}\,\mathbf{P}_{k-1}\,\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}. \end{aligned} \tag{3.20}$$

- The update step *is*

$$\begin{aligned} \mathbf{v}_k &= \mathbf{y}_k - \mathbf{H}_k\,\mathbf{m}_k^- \\ \mathbf{S}_k &= \mathbf{H}_k\,\mathbf{P}_k^-\,\mathbf{H}_k^T + \mathbf{R}_k \\ \mathbf{K}_k &= \mathbf{P}_k^-\,\mathbf{H}_k^T\,\mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T. \end{aligned} \tag{3.21}$$

The initial state has a given Gaussian prior distribution $\mathbf{x}_0 \sim \mathrm{N}(\mathbf{m}_0, \mathbf{P}_0)$, *which also defines the initial mean and covariance.*

The Kalman filter equations can be derived as follows:

1. By Lemma A.1 on page 119, the joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ given $\mathbf{y}_{1:k-1}$ is
$$\begin{aligned} p(\mathbf{x}_{k-1}, \mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1} \,|\, \mathbf{y}_{1:k-1}) \\ &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{A}_{k-1}\,\mathbf{x}_{k-1}, \mathbf{Q}_{k-1})\ \mathrm{N}(\mathbf{x}_{k-1} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1}) \\ &= \mathrm{N}\left(\begin{bmatrix} \mathbf{x}_{k-1} \\ \mathbf{x}_k \end{bmatrix} \,\Big|\, \mathbf{m}', \mathbf{P}' \right), \end{aligned} \tag{3.22}$$
where
$$\mathbf{m}' = \begin{pmatrix} \mathbf{m}_{k-1} \\ \mathbf{A}_{k-1}\,\mathbf{m}_{k-1} \end{pmatrix}, \quad \mathbf{P}' = \begin{pmatrix} \mathbf{P}_{k-1} & \mathbf{P}_{k-1}\,\mathbf{A}_{k-1}^T \\ \mathbf{A}_{k-1}\,\mathbf{P}_{k-1} & \mathbf{A}_{k-1}\,\mathbf{P}_{k-1}\,\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1} \end{pmatrix}. \tag{3.23}$$
and the marginal distribution of $\mathbf{x}_k$ is by Lemma A.2
$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) = \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-), \tag{3.24}$$
where
$$\mathbf{m}_k^- = \mathbf{A}_{k-1}\,\mathbf{m}_{k-1}, \qquad \mathbf{P}_k^- = \mathbf{A}_{k-1}\,\mathbf{P}_{k-1}\,\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}. \tag{3.25}$$

2. By Lemma A.1, the joint distribution of $\mathbf{y}_k$ and $\mathbf{x}_k$ is
$$\begin{aligned} p(\mathbf{x}_k, \mathbf{y}_k \,|\, \mathbf{y}_{1:k-1}) &= p(\mathbf{y}_k \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k-1}) \\ &= \mathrm{N}(\mathbf{y}_k \,|\, \mathbf{H}_k\,\mathbf{x}_k, \mathbf{R}_k)\ \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-) \\ &= \mathrm{N}\left(\begin{bmatrix} \mathbf{x}_k \\ \mathbf{y}_k \end{bmatrix} \,\Big|\, \mathbf{m}'', \mathbf{P}'' \right), \end{aligned} \tag{3.26}$$
where
$$\mathbf{m}'' = \begin{pmatrix} \mathbf{m}_k^- \\ \mathbf{H}_k\,\mathbf{m}_k^- \end{pmatrix}, \qquad \mathbf{P}'' = \begin{pmatrix} \mathbf{P}_k^- & \mathbf{P}_k^-\,\mathbf{H}_k^T \\ \mathbf{H}_k\,\mathbf{P}_k^- & \mathbf{H}_k\,\mathbf{P}_k^-\,\mathbf{H}_k^T + \mathbf{R}_k \end{pmatrix}. \tag{3.27}$$

3. By Lemma A.2 the conditional distribution of $\mathbf{x}_k$ is
$$\begin{aligned} p(\mathbf{x}_k \,|\, \mathbf{y}_k, \mathbf{y}_{1:k-1}) &= p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) \\ &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k), \end{aligned} \tag{3.28}$$
where
$$\begin{aligned} \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{P}_k^-\,\mathbf{H}_k^T(\mathbf{H}_k\,\mathbf{P}_k^-\,\mathbf{H}_k^T + \mathbf{R}_k)^{-1}[\mathbf{y}_k - \mathbf{H}_k\,\mathbf{m}_k^-] \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{P}_k^-\,\mathbf{H}_k^T\,(\mathbf{H}_k\,\mathbf{P}_k^-\,\mathbf{H}_k^T + \mathbf{R}_k)^{-1}\,\mathbf{H}_k\,\mathbf{P}_k^- \end{aligned} \tag{3.29}$$
which can be also written in form (3.21).

The functional form of the Kalman filter equations given here is not the only possible one. From a numerical stability point of view it would be better to work with matrix square roots of covariances instead of plain covariance matrices. The theory and details of implementation of this kind of methods is well covered, for example, in the book of Grewal and Andrews (2001).

Example 3.2 (Kalman filter for Gaussian random walk)**.** *Assume that we are observing measurements y_k of the Gaussian random walk model given in Example 3.1 and we want to estimate the state x_k at each time step. The information obtained up to time step $k-1$ is summarized by the Gaussian filtering density*

$$p(x_{k-1}\,|\,y_{1:k-1}) = \mathrm{N}(x_{k-1}\,|\,m_{k-1}, P_{k-1}). \tag{3.30}$$

The Kalman filter prediction and update equations are now given as

$$\begin{aligned} m_k^- &= m_{k-1} \\ P_k^- &= P_{k-1} + q \\ m_k &= m_k^- + \frac{P_k^-}{P_k^- + r}(y_k - m_k^-) \\ P_k &= P_k^- - \frac{(P_k^-)^2}{P_k^- + r}. \end{aligned} \tag{3.31}$$

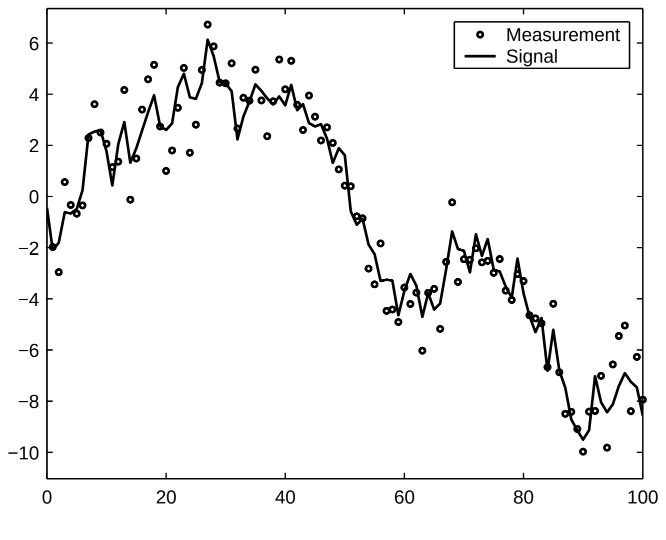


Figure 3.3: Simulated signal and measurements of the Kalman filtering example (Example 3.2).

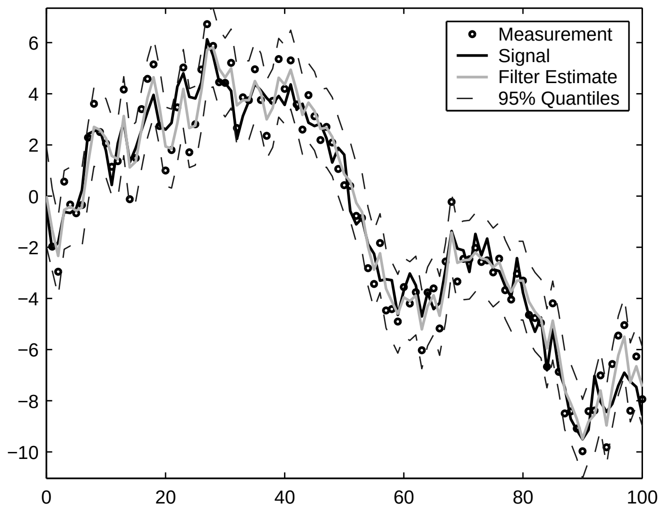


Figure 3.4: Signal, measurements and filtering estimate of the Kalman filtering example (Example 3.2).

3.2 Extended and Unscented Kalman Filtering

Often the dynamic and measurement processes in practical applications are not linear and the Kalman filter cannot be applied as such. However, often the filtering distributions of this kind of processes can be approximated with Gaussian distributions. In this section, four types of methods for forming the Gaussian approximations are considered, the Taylor series based extended Kalman filters (EKF), statistical linearization based statistically linearized filters (SLF), Fourier-Hermite expansion based Fourier-Hermite Kalman filters (FHKF), and unscented transform based unscented Kalman filters (UKF). Among these, UKF differs from the other filters in this section in the sense that it is not a series expansion based method per se — even though it was originally justified by considering a series expansion of the non-linear function.

3.2.1 Taylor Series Expansions

Consider the following transformation of a Gaussian random variable $\mathbf{x}$ into another random variable $\mathbf{y}$:

$$\begin{aligned} \mathbf{x} &\sim \mathrm{N}(\mathbf{m}, \mathbf{P}) \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}). \end{aligned} \tag{3.32}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^m$, and $\mathbf{g} : \mathbb{R}^n \mapsto \mathbb{R}^m$ is a general non-linear function. Formally, the probability density of the random variable $\mathbf{y}$ is[1] (see, e.g, Gelman

[1]This actually only applies to invertible $\mathbf{g}(\cdot)$, but it can be easily generalized to the non-invertible case.

et al., 1995)

$$p(\mathbf{y}) = |\mathbf{J}(\mathbf{y})| \; \mathrm{N}(\mathbf{g}^{-1}(\mathbf{y}) \,|\, \mathbf{m}, \mathbf{P}), \tag{3.33}$$

where $|\mathbf{J}(\mathbf{y})|$ is the determinant of the Jacobian matrix of the inverse transform $\mathbf{g}^{-1}(\mathbf{y})$. However, it is not generally possible to handle this distribution directly, because it is non-Gaussian for all but linear $\mathbf{g}$.

A first order Taylor series based Gaussian approximation to the distribution of $\mathbf{y}$ can be now formed as follows. If we let $\mathbf{x} = \mathbf{m} + \delta\mathbf{x}$, where $\delta\mathbf{x} \sim \mathrm{N}(\mathbf{0}, \mathbf{P})$, we can form the Taylor series expansion of the function $\mathbf{g}(\cdot)$ as follows:

$$\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{m}+\delta\mathbf{x}) = \mathbf{g}(\mathbf{m})+\mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\delta\mathbf{x}+\sum_i \frac{1}{2}\delta\mathbf{x}^T\,\mathbf{G}^{(i)}_{\mathbf{xx}}(\mathbf{m})\,\delta\mathbf{x}\,\mathbf{e}_i+\ldots \tag{3.34}$$

where $\mathbf{G}_{\mathbf{x}}(\mathbf{m})$ is the Jacobian matrix of $\mathbf{g}$ with elements

$$\left[\mathbf{G}_{\mathbf{x}}(\mathbf{m})\right]_{j,j'} = \left.\frac{\partial g_j(\mathbf{x})}{\partial x_{j'}}\right|_{\mathbf{x}=\mathbf{m}}. \tag{3.35}$$

and $\mathbf{G}^{(i)}_{\mathbf{xx}}(\mathbf{m})$ is the Hessian matrix of $g_i(\cdot)$ evaluated at $\mathbf{m}$:

$$\left[\mathbf{G}^{(i)}_{\mathbf{xx}}(\mathbf{m})\right]_{j,j'} = \left.\frac{\partial^2 g_i(\mathbf{x})}{\partial x_j\,\partial x_{j'}},\right|_{\mathbf{x}=\mathbf{m}}. \tag{3.36}$$

Also, $\mathbf{e}_i = (0\ \cdots\ 0\ 1\ 0\ \cdots\ 0)^T$ is a vector with 1 at position i and other elements are zero, that is, it is the unit vector in direction of the coordinate axis i.

The linear approximation can be obtained by approximating the function by the first two terms in the Taylor series:

$$\mathbf{g}(\mathbf{x}) \approx \mathbf{g}(\mathbf{m}) + \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\delta\mathbf{x}. \tag{3.37}$$

Computing the expected value with respect to $\mathbf{x}$ gives:

$$\begin{aligned}
\mathrm{E}[\mathbf{g}(\mathbf{x})] &\approx \mathrm{E}[\mathbf{g}(\mathbf{m}) + \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\delta\mathbf{x}] \\
&= \mathbf{g}(\mathbf{m}) + \mathbf{G}_{\mathbf{x}}(\mathbf{m})\ \mathrm{E}[\delta\mathbf{x}] \\
&= \mathbf{g}(\mathbf{m}).
\end{aligned} \tag{3.38}$$

The covariance can then be approximated as

$$\begin{aligned}
&\mathrm{E}\left[(\mathbf{g}(\mathbf{x}) - \mathrm{E}[\mathbf{g}(\mathbf{x})])\ (\mathbf{g}(\mathbf{x}) - \mathrm{E}[\mathbf{g}(\mathbf{x})])^T\right] \\
&\quad\approx \mathrm{E}\left[(\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{m}))\ (\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{m}))^T\right] \\
&\quad\approx \mathrm{E}\left[(\mathbf{g}(\mathbf{m}) + \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\delta\mathbf{x} - \mathbf{g}(\mathbf{m})])\ (\mathbf{g}(\mathbf{m}) + \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\delta\mathbf{x} - \mathbf{g}(\mathbf{m}))^T\right] \\
&\quad= \mathrm{E}\left[(\mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\delta\mathbf{x})\ (\mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\delta\mathbf{x})^T\right] \\
&\quad= \mathbf{G}_{\mathbf{x}}(\mathbf{m})\ \mathrm{E}\left[\delta\mathbf{x}\,\delta\mathbf{x}^T\right]\mathbf{G}^T_{\mathbf{x}}(\mathbf{m}) \\
&\quad= \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\mathbf{P}\,\mathbf{G}^T_{\mathbf{x}}(\mathbf{m}).
\end{aligned} \tag{3.39}$$

We are also often interested in the the joint covariance between the variables $\mathbf{x}$ and $\mathbf{y}$. Approximation of the joint covariance can be achieved by considering the augmented transformation

$$\tilde{\mathbf{g}}(\mathbf{x}) = \begin{pmatrix} \mathbf{x} \\ \mathbf{g}(\mathbf{x}) \end{pmatrix}. \tag{3.40}$$

The resulting mean and covariance are:

$$\begin{aligned} \mathrm{E}[\tilde{\mathbf{g}}(\mathbf{x})] &\approx \begin{pmatrix} \mathbf{m} \\ \mathbf{g}(\mathbf{m}) \end{pmatrix} \\ \mathrm{Cov}[\tilde{\mathbf{g}}(\mathbf{x})] &\approx \begin{pmatrix} \mathbf{I} \\ \mathbf{G}_{\mathbf{x}}(\mathbf{m}) \end{pmatrix} \mathbf{P} \begin{pmatrix} \mathbf{I} \\ \mathbf{G}_{\mathbf{x}}(\mathbf{m}) \end{pmatrix}^T \\ &= \begin{pmatrix} \mathbf{P} & \mathbf{P}\,\mathbf{G}_{\mathbf{x}}^T(\mathbf{m}) \\ \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\mathbf{P} & \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\mathbf{P}\,\mathbf{G}_{\mathbf{x}}^T(\mathbf{m}) \end{pmatrix}. \end{aligned} \tag{3.41}$$

In the derivation of the extended Kalman filter equations, we need a slightly more general transformation of the form

$$\begin{aligned} \mathbf{x} &\sim \mathrm{N}(\mathbf{m}, \mathbf{P}) \\ \mathbf{q} &\sim \mathrm{N}(\mathbf{0}, \mathbf{Q}) \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}) + \mathbf{q}, \end{aligned} \tag{3.42}$$

where $\mathbf{q}$ is independent of $\mathbf{x}$. The joint distribution of $\mathbf{x}$ and $\mathbf{y}$, as defined above, is now the same as in Equations (3.41) except that the covariance $\mathbf{Q}$ is added to the lower right block of the covariance matrix of $\tilde{\mathbf{g}}(\cdot)$. Thus we get the following algorithm:

Algorithm 3.2 (Linear approximation of an additive transform)**.** *The linear approximation based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$*, where* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_L \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_L \\ \mathbf{C}_L^T & \mathbf{S}_L \end{pmatrix} \right), \tag{3.43}$$

where

$$\begin{aligned} \boldsymbol{\mu}_L &= \mathbf{g}(\mathbf{m}) \\ \mathbf{S}_L &= \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\mathbf{P}\,\mathbf{G}_{\mathbf{x}}^T(\mathbf{m}) + \mathbf{Q} \\ \mathbf{C}_L &= \mathbf{P}\,\mathbf{G}_{\mathbf{x}}^T(\mathbf{m}), \end{aligned} \tag{3.44}$$

and $\mathbf{G}_{\mathbf{x}}(\mathbf{m})$ *is the Jacobian matrix of* $\mathbf{g}$ *with respect to* $\mathbf{x}$*, evaluated at* $\mathbf{x} = \mathbf{m}$ *with elements*

$$[\mathbf{G}_{\mathbf{x}}(\mathbf{m})]_{j,j'} = \left. \frac{\partial g_j(\mathbf{x})}{\partial x_{j'}} \right|_{\mathbf{x}=\mathbf{m}}. \tag{3.45}$$

Furthermore, in filtering models where the process noise is not additive, we often need to approximate transformations of the form

$$\begin{aligned} \mathbf{x} &\sim \mathrm{N}(\mathbf{m}, \mathbf{P}) \\ \mathbf{q} &\sim \mathrm{N}(\mathbf{0}, \mathbf{Q}) \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}, \mathbf{q}), \end{aligned} \tag{3.46}$$

where $\mathbf{x}$ and $\mathbf{q}$ are uncorrelated random variables. The mean and covariance can now be computed by substituting the augmented vector $(\mathbf{x}, \mathbf{q})$ to the vector $\mathbf{x}$ in Equation (3.41). The joint Jacobian matrix can then be written as $\mathbf{G}_{\mathbf{x},\mathbf{q}} = (\mathbf{G}_\mathbf{x} \ \mathbf{G}_\mathbf{q})$. Here $\mathbf{G}_\mathbf{q}$ is the Jacobian matrix of $\mathbf{g}(\cdot)$ with respect to $\mathbf{q}$ and both Jacobian matrices are evaluated at $\mathbf{x} = \mathbf{m}, \mathbf{q} = \mathbf{0}$. The approximations to the mean and covariance of the augmented transform as in Equation (3.41) are then given as

$$\begin{aligned} \mathrm{E}[\tilde{\mathbf{g}}(\mathbf{x}, \mathbf{q})] &\approx \mathbf{g}(\mathbf{m}, \mathbf{0}) \\ \mathrm{Cov}[\tilde{\mathbf{g}}(\mathbf{x}, \mathbf{q})] &\approx \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{G}_\mathbf{x}(\mathbf{m}) & \mathbf{G}_\mathbf{q}(\mathbf{m}) \end{pmatrix} \begin{pmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{G}_\mathbf{x}(\mathbf{m}) & \mathbf{G}_\mathbf{q}() \end{pmatrix}^T \\ &= \begin{pmatrix} \mathbf{P} & \mathbf{P}\,\mathbf{G}_\mathbf{x}^T(\mathbf{m}) \\ \mathbf{G}_\mathbf{x}(\mathbf{m})\,\mathbf{P} & \mathbf{G}_\mathbf{x}(\mathbf{m})\,\mathbf{P}\,\mathbf{G}_\mathbf{x}^T(\mathbf{m}) + \mathbf{G}_\mathbf{q}(\mathbf{m})\,\mathbf{Q}\,\mathbf{G}_\mathbf{q}^T(\mathbf{m}) \end{pmatrix}. \end{aligned} \tag{3.47}$$

The approximation above can be formulated as the following algorithm:

Algorithm 3.3 (Linear approximation of a non-additive transform)**.** *The linear approximation based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{q})$ *when* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_L \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_L \\ \mathbf{C}_L^T & \mathbf{S}_L \end{pmatrix} \right), \tag{3.48}$$

where

$$\begin{aligned} \boldsymbol{\mu}_L &= \mathbf{g}(\mathbf{m}) \\ \mathbf{S}_L &= \mathbf{G}_\mathbf{x}(\mathbf{m})\,\mathbf{P}\,\mathbf{G}_\mathbf{x}^T(\mathbf{m}) + \mathbf{G}_\mathbf{q}(\mathbf{m})\,\mathbf{Q}\,\mathbf{G}_\mathbf{q}^T(\mathbf{m}) \\ \mathbf{C}_L &= \mathbf{P}\,\mathbf{G}_\mathbf{x}^T(\mathbf{m}), \end{aligned} \tag{3.49}$$

and $\mathbf{G}_\mathbf{x}(\mathbf{m})$ *is the Jacobian matrix of* $\mathbf{g}$ *with respect to* $\mathbf{x}$*, evaluated at* $\mathbf{x} = \mathbf{m}, \mathbf{q} = \mathbf{0}$ *with elements*

$$[\mathbf{G}_\mathbf{x}(\mathbf{m})]_{j,j'} = \left. \frac{\partial g_j(\mathbf{x}, \mathbf{q})}{\partial x_{j'}} \right|_{\mathbf{x}=\mathbf{m},\mathbf{q}=\mathbf{0}}. \tag{3.50}$$

and $\mathbf{G}_\mathbf{q}(\mathbf{m})$ *is the corresponding Jacobian matrix with respect to* $\mathbf{q}$*:*

$$[\mathbf{G}_\mathbf{q}(\mathbf{m})]_{j,j'} = \left. \frac{\partial g_j(\mathbf{x}, \mathbf{q})}{\partial q_{j'}} \right|_{\mathbf{x}=\mathbf{m},\mathbf{q}=\mathbf{0}}. \tag{3.51}$$

In quadratic approximations, in addition to the first order terms also the second order terms in the Taylor series expansion of the non-linear function are retained:

Algorithm 3.4 (Quadratic approximation of an additive non-linear transform)**.** *The second order approximation is of the form*

$$\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix}\mathbf{m}\\ \boldsymbol{\mu}_Q\end{pmatrix}, \begin{pmatrix}\mathbf{P} & \mathbf{C}_Q\\ \mathbf{C}_Q^T & \mathbf{S}_Q\end{pmatrix}\right), \tag{3.52}$$

where the parameters are

$$\begin{aligned}
\boldsymbol{\mu}_Q &= \mathbf{g}(\mathbf{m}) + \frac{1}{2}\sum_i \mathbf{e}_i \,\mathrm{tr}\left\{\mathbf{G}_{\mathbf{xx}}^{(i)}(\mathbf{m})\,\mathbf{P}\right\}\\
\mathbf{S}_Q &= \mathbf{G}_{\mathbf{x}}(\mathbf{m})\,\mathbf{P}\,\mathbf{G}_{\mathbf{x}}^T(\mathbf{m}) + \frac{1}{2}\sum_{i,i'} \mathbf{e}_i\,\mathbf{e}_{i'}^T \,\mathrm{tr}\left\{\mathbf{G}_{\mathbf{xx}}^{(i)}(\mathbf{m})\,\mathbf{P}\,\mathbf{G}_{\mathbf{xx}}^{(i')}(\mathbf{m})\,\mathbf{P}\right\}\\
\mathbf{C}_Q &= \mathbf{P}\,\mathbf{G}_{\mathbf{x}}^T(\mathbf{m}),
\end{aligned} \tag{3.53}$$

and $\mathbf{G}_{\mathbf{x}}(\mathbf{m})$ is the Jacobian matrix (3.45)*, and $\mathbf{G}_{\mathbf{xx}}^{(i)}(\mathbf{m})$ is the Hessian matrix of $g_i(\cdot)$ evaluated at $\mathbf{m}$:*

$$\left[\mathbf{G}_{\mathbf{xx}}^{(i)}(\mathbf{m})\right]_{j,j'} = \left.\frac{\partial^2 g_i(\mathbf{x})}{\partial x_j\,\partial x_{j'}}\right|_{\mathbf{x}=\mathbf{m}}, \tag{3.54}$$

where $\mathbf{e}_i = (0\ \cdots\ 0\ 1\ 0\ \cdots\ 0)^T$ is a vector with 1 at position i and other elements are zero, that is, it is the unit vector in direction of the coordinate axis i.

3.2.2 Extended Kalman Filter (EKF)

The extended Kalman filter (EKF) (see, e.g., Jazwinski, 1970; Maybeck, 1982a; Bar-Shalom et al., 2001; Grewal and Andrews, 2001) is an extension of the Kalman filter to non-linear optimal filtering problems. If process and measurement noises can be assumed to be additive, the EKF model can be written as

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{q}_{k-1}\\
\mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k) + \mathbf{r}_k,
\end{aligned} \tag{3.55}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state, $\mathbf{y}_k \in \mathbb{R}^m$ is the measurement, $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ is the Gaussian process noise, $\mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k)$ is the Gaussian measurement noise, $\mathbf{f}(\cdot)$ is the dynamic model function and $\mathbf{h}(\cdot)$ is the measurement model function. The functions $\mathbf{f}$ and $\mathbf{h}$ can also depend on the step number k, but for notational convenience, this dependence has not been explicitly denoted.

The idea of the extended Kalman filter is to form Gaussian approximations

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) \approx \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k) \tag{3.56}$$

to the filtering densities. In EKF this is done by utilizing linear approximations to the non-linearities and the result is the following algorithm.

Algorithm 3.5 (Extended Kalman filter I)**.** *The prediction and update steps of the first order additive noise extended Kalman filter (EKF) are:*

- *Prediction:*

$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{f}(\mathbf{m}_{k-1}) \\ \mathbf{P}_k^- &= \mathbf{F}_\mathbf{x}(\mathbf{m}_{k-1})\,\mathbf{P}_{k-1}\,\mathbf{F}_\mathbf{x}^T(\mathbf{m}_{k-1}) + \mathbf{Q}_{k-1}. \end{aligned} \tag{3.57}$$

- *Update:*

$$\begin{aligned} \mathbf{v}_k &= \mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-) \\ \mathbf{S}_k &= \mathbf{H}_\mathbf{x}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\,\mathbf{H}_\mathbf{x}^T(\mathbf{m}_k^-) + \mathbf{R}_k \\ \mathbf{K}_k &= \mathbf{P}_k^-\,\mathbf{H}_\mathbf{x}^T(\mathbf{m}_k^-)\,\mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T. \end{aligned} \tag{3.58}$$

These filtering equations can be derived by repeating the same steps as in the derivation of the Kalman filter in Section 3.1.3 and by applying Taylor series approximations on the appropriate steps:

1. The joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ is non-Gaussian, but we can form a Gaussian approximation to it by applying the approximation Algorithm 3.2 to the function

$$\mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{q}_{k-1}, \tag{3.59}$$

which results in the Gaussian approximation

$$p(\mathbf{x}_{k-1}, \mathbf{x}_k, \mid \mathbf{y}_{1:k-1}) \approx \mathrm{N}\left(\begin{bmatrix} \mathbf{x}_{k-1} \\ \mathbf{x}_k \end{bmatrix} \,\middle|\, \mathbf{m}', \mathbf{P}' \right), \tag{3.60}$$

where

$$\begin{aligned} \mathbf{m}' &= \begin{pmatrix} \mathbf{m}_{k-1} \\ \mathbf{f}(\mathbf{m}_{k-1}) \end{pmatrix} \\ \mathbf{P}' &= \begin{pmatrix} \mathbf{P}_{k-1} & \mathbf{P}_{k-1}\,\mathbf{F}_\mathbf{x}^T \\ \mathbf{F}_\mathbf{x}\,\mathbf{P}_{k-1} & \mathbf{F}_\mathbf{x}\,\mathbf{P}_{k-1}\,\mathbf{F}_\mathbf{x}^T + \mathbf{Q}_{k-1} \end{pmatrix}, \end{aligned} \tag{3.61}$$

and the Jacobian matrix $\mathbf{F}_\mathbf{x}$ of $\mathbf{f}(\mathbf{x})$ is evaluated at $\mathbf{x} = \mathbf{m}_{k-1}$. The marginal mean and covariance of $\mathbf{x}_k$ are thus

$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{f}(\mathbf{m}_{k-1}) \\ \mathbf{P}_k^- &= \mathbf{F}_\mathbf{x}\,\mathbf{P}_{k-1}\,\mathbf{F}_\mathbf{x}^T + \mathbf{Q}_{k-1}. \end{aligned} \tag{3.62}$$

2. The joint distribution of $\mathbf{y}_k$ and $\mathbf{x}_k$ is also non-Gaussian, but we can again approximate it by applying Algorithm 3.2 to the function
$$\mathbf{h}(\mathbf{x}_k) + \mathbf{r}_k. \tag{3.63}$$
We get the approximation
$$p(\mathbf{x}_k, \mathbf{y}_k \,|\, \mathbf{y}_{1:k-1}) \approx \mathrm{N}\left(\begin{bmatrix} \mathbf{x}_k \\ \mathbf{y}_k \end{bmatrix} \,\middle|\, \mathbf{m}'', \mathbf{P}'' \right), \tag{3.64}$$
where
$$\mathbf{m}'' = \begin{pmatrix} \mathbf{m}_k^- \\ \mathbf{h}(\mathbf{m}_k^-) \end{pmatrix}, \qquad \mathbf{P}'' = \begin{pmatrix} \mathbf{P}_k^- & \mathbf{P}_k^- \,\mathbf{H}_{\mathbf{x}}^T \\ \mathbf{H}_{\mathbf{x}}\,\mathbf{P}_k^- & \mathbf{H}_{\mathbf{x}}\,\mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T + \mathbf{R}_k \end{pmatrix}, \tag{3.65}$$
and the Jacobian matrix $\mathbf{H}_{\mathbf{x}}$ of $\mathbf{h}(\mathbf{x})$ is evaluated at $\mathbf{x} = \mathbf{m}_k^-$.
3. By Lemma A.2 the conditional distribution of $\mathbf{x}_k$ is approximately
$$p(\mathbf{x}_k \,|\, \mathbf{y}_k, \mathbf{y}_{1:k-1}) \approx \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k), \tag{3.66}$$
where
$$\begin{aligned} \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T(\mathbf{H}_{\mathbf{x}}\,\mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T + \mathbf{R}_k)^{-1}[\mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-)] \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T\,(\mathbf{H}_{\mathbf{x}}\,\mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T + \mathbf{R}_k)^{-1}\,\mathbf{H}_{\mathbf{x}}\,\mathbf{P}_k^-. \end{aligned} \tag{3.67}$$

A more general non-additive noise EKF filtering model can be written as
$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) \\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k), \end{aligned} \tag{3.68}$$
where $\mathbf{q}_{k-1} \sim \mathrm{N}(0, \mathbf{Q}_{k-1})$ and $\mathbf{r}_k \sim \mathrm{N}(0, \mathbf{R}_k)$ are the Gaussian process and measurement noises, respectively. Again, the functions $\mathbf{f}$ and $\mathbf{h}$ can also depend on the step number k.

Algorithm 3.6 (Extended Kalman filter II)**.** *The prediction and update steps of the (first order) extended Kalman filter (EKF) in the non-additive noise case are:*

- *Prediction:*
$$\begin{aligned} \mathbf{m}_k^- &= \mathbf{f}(\mathbf{m}_{k-1}, \mathbf{0}) \\ \mathbf{P}_k^- &= \mathbf{F}_{\mathbf{x}}(\mathbf{m}_{k-1})\,\mathbf{P}_{k-1}\,\mathbf{F}_{\mathbf{x}}^T(\mathbf{m}_{k-1}) + \mathbf{F}_{\mathbf{q}}(\mathbf{m}_{k-1})\,\mathbf{Q}_{k-1}\,\mathbf{F}_{\mathbf{q}}^T(\mathbf{m}_{k-1}). \end{aligned} \tag{3.69}$$
- *Update:*
$$\begin{aligned} \mathbf{v}_k &= \mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-, \mathbf{0}) \\ \mathbf{S}_k &= \mathbf{H}_{\mathbf{x}}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}_k^-) + \mathbf{H}_{\mathbf{r}}(\mathbf{m}_k^-)\,\mathbf{R}_k\,\mathbf{H}_{\mathbf{r}}^T(\mathbf{m}_k^-) \\ \mathbf{K}_k &= \mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}_k^-)\,\mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T. \end{aligned} \tag{3.70}$$

where the matrices $\mathbf{F}_{\mathbf{x}}(\mathbf{m})$*,* $\mathbf{F}_{\mathbf{q}}(\mathbf{m})$*,* $\mathbf{H}_{\mathbf{x}}(\mathbf{m})$*, and* $\mathbf{H}_{\mathbf{r}}(\mathbf{m})$*, are the Jacobian matrices of* $\mathbf{f}$ *and* $\mathbf{h}$ *with respect to state and noise, with elements*

$$[\mathbf{F}_{\mathbf{x}}(\mathbf{m})]_{j,j'} = \left.\frac{\partial f_j(\mathbf{x},\mathbf{q})}{\partial x_{j'}}\right|_{\mathbf{x}=\mathbf{m},\mathbf{q}=\mathbf{0}} \tag{3.71}$$

$$[\mathbf{F}_{\mathbf{q}}(\mathbf{m})]_{j,j'} = \left.\frac{\partial f_j(\mathbf{x},\mathbf{q})}{\partial q_{j'}}\right|_{\mathbf{x}=\mathbf{m},\mathbf{q}=\mathbf{0}} \tag{3.72}$$

$$[\mathbf{H}_{\mathbf{x}}(\mathbf{m})]_{j,j'} = \left.\frac{\partial h_j(\mathbf{x},\mathbf{r})}{\partial x_{j'}}\right|_{\mathbf{x}=\mathbf{m},\mathbf{r}=\mathbf{0}} \tag{3.73}$$

$$[\mathbf{H}_{\mathbf{r}}(\mathbf{m})]_{j,j'} = \left.\frac{\partial h_j(\mathbf{x},\mathbf{r})}{\partial r_{j'}}\right|_{\mathbf{x}=\mathbf{m},\mathbf{r}=\mathbf{0}}. \tag{3.74}$$

These filtering equations can be derived by repeating the same steps as in the derivation of the extended Kalman filter above, but instead of using Algorithm 3.2, we use Algorithm 3.3 for computing the approximations.

The advantage of EKF over other non-linear filtering methods is its relative simplicity compared to its performance. Linearization is very common engineering way of constructing approximations to non-linear systems and thus it is very easy to understand and apply. A disadvantage is that because it is based on a local linear approximation, it will not work in problems with considerable non-linearities. Also the filtering model is restricted in the sense that only Gaussian noise processes are allowed and thus the model cannot contain, for example, discrete valued random variables. The Gaussian restriction also prevents handling of hierarchical models or other models where significantly non-Gaussian distribution models would be needed.

The EKF also requires the measurement model and the dynamic model functions to be differentiable. This as such might be a restriction, but in some cases it might also be simply impossible to compute the required Jacobian matrices, which renders the usage of EKF impossible. And even when the Jacobian matrices exist and could be computed, the actual computation and programming of Jacobian matrices can be quite error prone and hard to debug.

In so called second order EKF the non-linearity is approximated by retaining the second order terms in the Taylor series expansion as in Algorithm 3.4:

Algorithm 3.7 (Extended Kalman filter III)**.** *The prediction and update steps of the second order extended Kalman filter (in additive noise case) are:*

- *Prediction:*

$$\begin{aligned}
\mathbf{m}_k^- &= \mathbf{f}(\mathbf{m}_{k-1}) + \frac{1}{2}\sum_i \mathbf{e}_i \, \operatorname{tr}\left\{\mathbf{F}_{\mathbf{xx}}^{(i)}(\mathbf{m}_{k-1})\,\mathbf{P}_{k-1}\right\} \\
\mathbf{P}_k^- &= \mathbf{F}_{\mathbf{x}}(\mathbf{m}_{k-1})\,\mathbf{P}_{k-1}\,\mathbf{F}_{\mathbf{x}}^T(\mathbf{m}_{k-1}) \\
&\quad + \frac{1}{2}\sum_{i,i'} \mathbf{e}_i\,\mathbf{e}_{i'}^T \operatorname{tr}\left\{\mathbf{F}_{\mathbf{xx}}^{(i)}(\mathbf{m}_{k-1})\mathbf{P}_{k-1}\mathbf{F}_{\mathbf{xx}}^{(i')}(\mathbf{m}_{k-1})\mathbf{P}_{k-1}\right\} \\
&\quad + \mathbf{Q}_{k-1}.
\end{aligned} \tag{3.75}$$

- *Update:*

$$\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-) - \frac{1}{2}\sum_i \mathbf{e}_i \, \operatorname{tr}\left\{\mathbf{H}_{\mathbf{xx}}^{(i)}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\right\} \\
\mathbf{S}_k &= \mathbf{H}_{\mathbf{x}}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}_k^-) \\
&\quad + \frac{1}{2}\sum_{i,i'} \mathbf{e}_i\,\mathbf{e}_{i'}^T \, \operatorname{tr}\left\{\mathbf{H}_{\mathbf{xx}}^{(i)}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\,\mathbf{H}_{\mathbf{xx}}^{(i')}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\right\} + \mathbf{R}_k \\
\mathbf{K}_k &= \mathbf{P}_k^-\,\mathbf{H}_{\mathbf{x}}^T(\mathbf{m}_k^-)\,\mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T,
\end{aligned} \tag{3.76}$$

where the matrices $\mathbf{F}_{\mathbf{x}}(\mathbf{m})$ *and* $\mathbf{H}_{\mathbf{x}}(\mathbf{m})$ *are given by the Equations* (3.71) *and* (3.73). *The matrices* $\mathbf{F}_{\mathbf{xx}}^{(i)}(\mathbf{m})$ *and* $\mathbf{H}_{\mathbf{xx}}^{(i)}(\mathbf{m})$ *are the Hessian matrices of* f_i *and* h_i *respectively:*

$$\left[\mathbf{F}_{\mathbf{xx}}^{(i)}(\mathbf{m})\right]_{j,j'} = \left.\frac{\partial^2 f_i(\mathbf{x})}{\partial x_j\,\partial x_{j'}}\right|_{\mathbf{x}=\mathbf{m}} \tag{3.77}$$

$$\left[\mathbf{H}_{\mathbf{xx}}^{(i)}(\mathbf{m})\right]_{j,j'} = \left.\frac{\partial^2 h_i(\mathbf{x})}{\partial x_j\,\partial x_{j'}}\right|_{\mathbf{x}=\mathbf{m}}. \tag{3.78}$$

The non-additive version can be derived in analogous manner, but due to its complicated appearance, it is not presented here.

3.2.3 Statistical Linearization

In statistically linearized filter (Gelb, 1974) the first order Taylor series approximation used in the first order EKF is replaced by statistical linearization. Recall the transformation problem considered in Section 3.2.1, which was stated as

$$\begin{aligned}
\mathbf{x} &\sim \mathrm{N}(\mathbf{m}, \mathbf{P}) \\
\mathbf{y} &= \mathbf{g}(\mathbf{x}).
\end{aligned}$$

In statistical linearization we form a linear approximation to the transformation as follows:

$$\mathbf{g}(\mathbf{x}) \approx \mathbf{b} + \mathbf{A}\,\delta\mathbf{x}, \tag{3.79}$$

where $\delta\mathbf{x} = \mathbf{x} - \mathbf{m}$, such that the mean squared error is minimized:

$$\mathrm{MSE}(\mathbf{b},\mathbf{A}) = \mathrm{E}[(\mathbf{g}(\mathbf{x}) - \mathbf{b} - \mathbf{A}\,\delta\mathbf{x})^T (\mathbf{g}(\mathbf{x}) - \mathbf{b} - \mathbf{A}\,\delta\mathbf{x})]. \tag{3.80}$$

Setting derivatives with respect to $\mathbf{b}$ and $\mathbf{A}$ zero gives

$$\begin{aligned} \mathbf{b} &= \mathrm{E}[\mathbf{g}(\mathbf{x})] \\ \mathbf{A} &= \mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]\,\mathbf{P}^{-1}. \end{aligned} \tag{3.81}$$

In this approximation to the transform $\mathbf{g}(\mathbf{x})$, $\mathbf{b}$ is now exactly the mean and the approximate covariance is given as

$$\begin{aligned} &\mathrm{E}[(\mathbf{g}(\mathbf{x}) - \mathrm{E}[\mathbf{g}(\mathbf{x})])\;(\mathbf{g}(\mathbf{x}) - \mathrm{E}[\mathbf{g}(\mathbf{x})])^T] \\ &\quad\approx \mathbf{A}\,\mathbf{P}\,\mathbf{A}^T \\ &\quad= \mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]\,\mathbf{P}^{-1}\;\mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]^T. \end{aligned} \tag{3.82}$$

We may now apply this approximation to the augmented function $\tilde{\mathbf{g}}(\mathbf{x}) = (\mathbf{x}, \mathbf{g}(\mathbf{x}))$ in Equation (3.40) of Section 3.2.1, where we get the approximation

$$\begin{aligned} \mathrm{E}[\tilde{\mathbf{g}}(\mathbf{x})] &\approx \begin{pmatrix} \mathbf{m} \\ \mathrm{E}[\mathbf{g}(\mathbf{x})] \end{pmatrix} \\ \mathrm{Cov}[\tilde{\mathbf{g}}(\mathbf{x})] &\approx \begin{pmatrix} \mathbf{P} & \mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}]^T \\ \mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T] & \mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]\,\mathbf{P}^{-1}\;\mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]^T \end{pmatrix}. \end{aligned} \tag{3.83}$$

We now get the following algorithm corresponding to Algorithm 3.2:

Algorithm 3.8 (Statistically linearized approximation of an additive transform)**.** *The statistical linearization based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ *where* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_S \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_S \\ \mathbf{C}_S^T & \mathbf{S}_S \end{pmatrix} \right), \tag{3.84}$$

where

$$\begin{aligned} \boldsymbol{\mu}_S &= \mathrm{E}[\mathbf{g}(\mathbf{x})] \\ \mathbf{S}_S &= \mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]\,\mathbf{P}^{-1}\;\mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]^T + \mathbf{Q} \\ \mathbf{C}_S &= \mathrm{E}[\mathbf{g}(\mathbf{x})\,\delta\mathbf{x}^T]^T. \end{aligned} \tag{3.85}$$

The expectations are taken with respect to the distribution of $\mathbf{x}$*.*

Applying the same approximation with $(\mathbf{x},\mathbf{q})$ in place of $\mathbf{x}$ we obtain the following mean and covariance:

$$
\begin{aligned}
\mathrm{E}[\tilde{\mathbf{g}}(\mathbf{x},\mathbf{q})] &\approx \begin{pmatrix} \mathbf{m} \\ \mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})] \end{pmatrix} \\
\mathrm{Cov}[\tilde{\mathbf{g}}(\mathbf{x},\mathbf{q})] &\approx \begin{pmatrix} \mathbf{P} & \mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\delta\mathbf{x}^T]^T \\ \mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\delta\mathbf{x}^T] & \begin{array}{l}\mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\delta\mathbf{x}^T]\,\mathbf{P}^{-1}\,\mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\delta\mathbf{x}^T]^T \\ +\,\mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\mathbf{q}^T]\,\mathbf{Q}^{-1}\,\mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\mathbf{q}^T]^T\end{array} \end{pmatrix}
\end{aligned}
\tag{3.86}
$$

Thus we get the following algorithm for non-additive transform:

Algorithm 3.9 (Statistically linearized approximation of a non-additive transform)**.** *The statistical linearization based Gaussian approximation to the joint distribution of $\mathbf{x}$ and the transformed random variable $\mathbf{y} = \mathbf{g}(\mathbf{x},\mathbf{q})$ when $\mathbf{x} \sim \mathrm{N}(\mathbf{m},\mathbf{P})$ and $\mathbf{q} \sim \mathrm{N}(\mathbf{0},\mathbf{Q})$ is given as*

$$
\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_S \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_S \\ \mathbf{C}_S^T & \mathbf{S}_S \end{pmatrix} \right), \tag{3.87}
$$

where

$$
\begin{aligned}
\boldsymbol{\mu}_S &= \mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})] \\
\mathbf{S}_S &= \mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\delta\mathbf{x}^T]\,\mathbf{P}^{-1}\,\mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\delta\mathbf{x}^T]^T + \mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\mathbf{q}^T]\,\mathbf{Q}^{-1}\,\mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\mathbf{q}^T]^T \\
\mathbf{C}_S &= \mathrm{E}[\mathbf{g}(\mathbf{x},\mathbf{q})\,\delta\mathbf{x}^T]^T.
\end{aligned}
\tag{3.88}
$$

The expectations are taken with respect to the variables $\mathbf{x}$ and $\mathbf{q}$.

If the function $\mathbf{g}(\mathbf{x})$ is differentiable, it is possible to use the following well known property of Gaussian random variables for simplifying the expressions:

$$
\mathrm{E}[\mathbf{g}(\mathbf{x})\,(\mathbf{x}-\mathbf{m})^T] = \mathrm{E}[\mathbf{G}_{\mathbf{x}}(\mathbf{x})]\,\mathbf{P}, \tag{3.89}
$$

where $\mathrm{E}[\cdot]$ denotes the expected value with respect to $\mathbf{x} \sim \mathrm{N}(\mathbf{m},\mathbf{P})$, and $\mathbf{G}_{\mathbf{x}}(\mathbf{x})$ is the Jacobian matrix of $\mathbf{g}(\mathbf{x})$. The statistical linearization equations then reduce to the same form as Taylor series based linearization, except that instead of the Jacobians we have the expected values of the Jacobians (see exercises). Algorithm 3.8 can be then written in the following form:

Algorithm 3.10 (Statistically linearized approximation of an additive transform II)**.** *The statistical linearization based Gaussian approximation to the joint distribution of $\mathbf{x}$ and the transformed random variable $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ where $\mathbf{x} \sim \mathrm{N}(\mathbf{m},\mathbf{P})$ and $\mathbf{q} \sim \mathrm{N}(\mathbf{0},\mathbf{Q})$, can be written as*

$$
\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_S \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_S \\ \mathbf{C}_S^T & \mathbf{S}_S \end{pmatrix} \right), \tag{3.90}
$$

where

$$\begin{aligned}
\boldsymbol{\mu}_S &= \mathrm{E}[\mathbf{g}(\mathbf{x})] \\
\mathbf{S}_S &= \mathrm{E}[\mathbf{G}_\mathbf{x}(\mathbf{x})]\,\mathbf{P}\;\mathrm{E}[\mathbf{G}_\mathbf{x}(\mathbf{x})]^T + \mathbf{Q} \\
\mathbf{C}_S &= \mathbf{P}\;\mathrm{E}[\mathbf{G}_\mathbf{x}(\mathbf{x})]^T,
\end{aligned} \tag{3.91}$$

and $\mathbf{G}_\mathbf{x}(\mathbf{x})$ *is the Jacobian matrix of* $\mathbf{g}$. *The expectations are taken with respect to the distribution of* $\mathbf{x}$.

Note that we actually only need to compute the expectation $\mathrm{E}[\mathbf{g}(\mathbf{x})]$, because if we know the function

$$\boldsymbol{\mu}_S(\mathbf{m}) = \mathrm{E}[\mathbf{g}(\mathbf{x})], \tag{3.92}$$

where $\mathrm{E}[\cdot]$ denotes the expected value with respect to $\mathrm{N}(\mathbf{x} \mid \mathbf{m}, \mathbf{P})$, then

$$\frac{\partial \boldsymbol{\mu}_S(\mathbf{m})}{\partial \mathbf{m}} = \mathrm{E}[\mathbf{G}_\mathbf{x}(\mathbf{x})]. \tag{3.93}$$

3.2.4 Statistically Linearized Filter

Statistically linearized filter (SLF) (Gelb, 1974) or quasi-linear filter (Stengel, 1994) is a Gaussian approximation based filter which can be applied to the same kind of models as EKF, that is, to models of the form (3.55) or (3.68). The filter is similar to EKF, except that statistical linearizations in Algorithms 3.8, 3.9 and 3.10 are used instead of the Taylor series approximations.

Algorithm 3.11 (Statistically linearized filter I)**.** *The prediction and update steps of the additive noise statistically linearized (Kalman) filter are:*

- *Prediction:*

$$\begin{aligned}
\mathbf{m}_k^- &= \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1})] \\
\mathbf{P}_k^- &= \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1})\,\delta\mathbf{x}_{k-1}^T]\,\mathbf{P}_{k-1}^{-1}\;\mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1})\,\delta\mathbf{x}_{k-1}^T]^T + \mathbf{Q}_{k-1},
\end{aligned} \tag{3.94}$$

  where $\delta\mathbf{x}_{k-1} = \mathbf{x}_{k-1} - \mathbf{m}_{k-1}$ *and the expectations are taken with respect to the variable* $\mathbf{x}_{k-1} \sim \mathrm{N}(\mathbf{m}_{k-1}, \mathbf{P}_{k-1})$.

- *Update:*

$$\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \mathrm{E}[\mathbf{h}(\mathbf{x}_k)] \\
\mathbf{S}_k &= \mathrm{E}[\mathbf{h}(\mathbf{x}_k)\,\delta\mathbf{x}_k^T]\,(\mathbf{P}_k^-)^{-1}\;\mathrm{E}[\mathbf{h}(\mathbf{x}_k)\,\delta\mathbf{x}_k^T]^T + \mathbf{R}_k \\
\mathbf{K}_k &= \mathrm{E}[\mathbf{h}(\mathbf{x}_k)\,\delta\mathbf{x}_k^T]^T\,\mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^T,
\end{aligned} \tag{3.95}$$

  where the expectations are taken with respect to the variable $\mathbf{x}_k \sim \mathrm{N}(\mathbf{m}_k^-, \mathbf{P}_k^-)$.

The above filter can also be rewritten using the expectations of Jacobians by using the Algorithm 3.10 instead of 3.8 (see exercises).

Algorithm 3.12 (Statistically linearized filter II)**.** *The prediction and update steps of the non-additive statistically linearized (Kalman) filter are:*

- *Prediction:*

$$\begin{aligned}\mathbf{m}_k^- &= \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})] \\ \mathbf{P}_k^- &= \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})\, \delta\mathbf{x}_{k-1}^T]\, \mathbf{P}_{k-1}^{-1}\ \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})\, \delta\mathbf{x}_{k-1}^T]^T \\ &\quad + \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})\, \mathbf{q}_{k-1}^T]\, \mathbf{Q}_{k-1}^{-1}\ \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})\, \mathbf{q}_{k-1}^T]^T,\end{aligned} \tag{3.96}$$

  where $\delta\mathbf{x}_{k-1} = \mathbf{x}_{k-1} - \mathbf{m}_{k-1}$ *and the expectations are taken with respect to the variables* $\mathbf{x}_{k-1} \sim \mathrm{N}(\mathbf{m}_{k-1}, \mathbf{P}_{k-1})$ *and* $\mathbf{q}_{k-1} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_{k-1})$.

- *Update:*

$$\begin{aligned}\mathbf{v}_k &= \mathbf{y}_k - \mathrm{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k)] \\ \mathbf{S}_k &= \mathrm{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k)\, \delta\mathbf{x}_k^T]\, (\mathbf{P}_k^-)^{-1}\ \mathrm{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k)\, \delta\mathbf{x}_k^T]^T \\ &\quad + \mathrm{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k)\, \mathbf{r}_k^T]\, \mathbf{R}_k^{-1}\ \mathrm{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k)\, \mathbf{r}_k^T]^T \\ \mathbf{K}_k &= \mathrm{E}[\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k)\, \delta\mathbf{x}_k^T]^T\, \mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, \mathbf{v}_k \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T,\end{aligned} \tag{3.97}$$

  where the expectations are taken with respect to the variables $\mathbf{x}_k \sim \mathrm{N}(\mathbf{m}_k^-, \mathbf{P}_k^-)$ *and* $\mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k)$.

Both the filters above can be derived by following the derivation of the EKF in Section 3.2.2 and by utilizing the statistical linearization approximations instead of the linear approximations on the appropriate steps.

The advantage of SLF over EKF is that it is a more global approximation than EKF, because the linearization is not only based on the local region around the mean but on a whole range of function values. The non-linearities also do not have to be differentiable nor do we need to derive their Jacobian matrices. However, if the non-linearities are differentiable, then we can use the Gaussian random variable property (3.89) for rewriting the equations in EKF-like form. The clear disadvantage of SLF over EKF is that certain expected values of the non-linear functions have to be computed in closed form. Naturally, it is not possible for all functions. Fortunately, the expected values involved are of such type that one is likely to find many of them tabulated in older physics and control engineering books (see, e.g. Gelb and Vander Velde, 1968).

The statistically linearized filter (SLF) is a special case of the Fourier-Hermite Kalman filter (FHKF), when the first order truncation of the series is used (Sarmavuori and Särkkä, 2012). Many of the sigma-point methods can also be interpreted as approximations to the Fourier-Hermite Kalman filters and statistically linearized filters (cf. Van der Merwe and Wan, 2003; Särkkä and Hartikainen, 2010b; Sarmavuori and Särkkä, 2012).

3.2.5 Unscented Transform

The *unscented transform* (UT) (Julier and Uhlmann, 1995; Julier et al., 2000) is a relatively recent numerical method that can be also used for approximating the joint distribution of random variables $\mathbf{x}$ and $\mathbf{y}$ defined as

$$\begin{aligned}\mathbf{x} &\sim \mathrm{N}(\mathbf{m}, \mathbf{P}) \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}).\end{aligned}$$

However, the philosophy in UT differs from linearization and statistical linearization in the sense that it tries to directly approximate the mean and covariance of the target distribution instead of trying to approximate the non-linear function (Julier and Uhlmann, 1995).

The idea of UT is to deterministically choose a fixed number of sigma-points that capture the mean and covariance of the original distribution of $\mathbf{x}$ exactly. These sigma-points are then propagated through the non-linearity and the mean and covariance of the transformed variable are estimated from them. Note that although the unscented transform resembles Monte Carlo estimation the approaches are significantly different, because in UT the sigma points are selected deterministically (Julier and Uhlmann, 2004). The difference between linear approximation and UT is illustrated in Figures 3.5, 3.6 and 3.7.

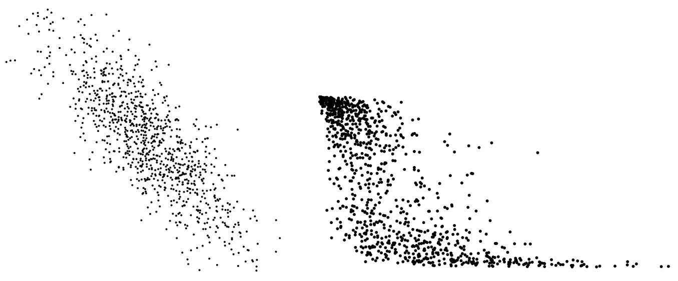


Figure 3.5: *Example of applying a non-linear transformation to a random variable on the left, which results in the random variable on the right.*

The *unscented transform* forms the Gaussian approximation[2] with the following procedure:

[2]Note that this Gaussianity assumption is one interpretation, but unscented transform can also be applied without the Gaussian assumption. However, because the assumption makes Bayesian interpretation of UT much easier, we shall use it here.

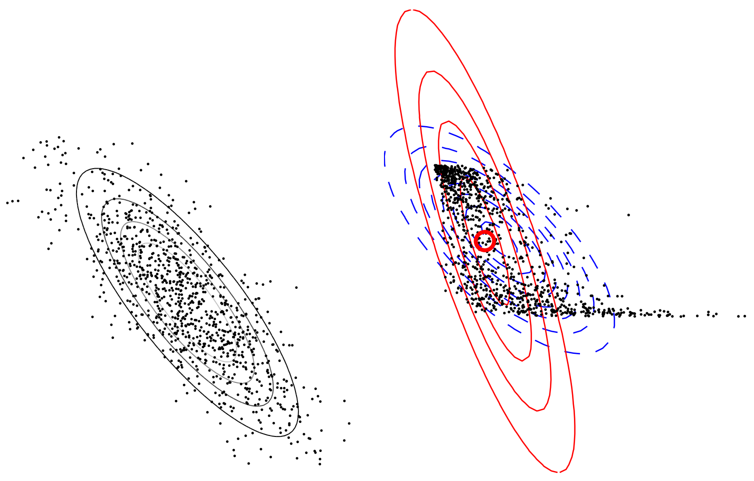


Figure 3.6: *Illustration of linearization based (EKF) approximation to the transformation in Figure 3.5. The Gaussian approximation is formed by calculating the curvature at the mean, which results in a bad approximation further away from the mean. The covariance of the true distribution is presented by the blue dotted line and the red solid line is the approximation.*

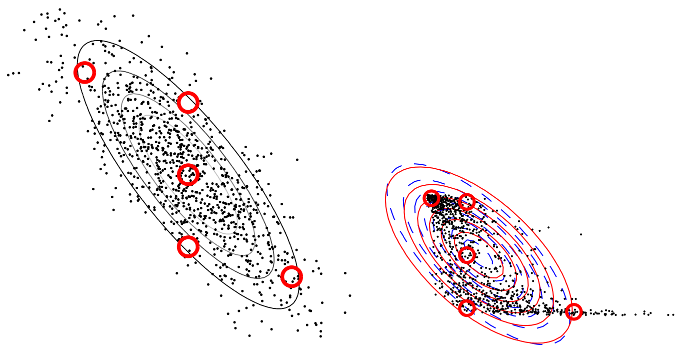


Figure 3.7: *Illustration of unscented transform based (UKF) approximation to the transformation in Figure 3.5. The Gaussian approximation is formed by propagating the sigma points through the non-linearity and the mean and covariance are estimated from the transformed sigma points. The covariance of the true distribution is presented by the blue dotted line and the red solid line is the approximation.*

1. Form a set of $2n + 1$ sigma points as follows:

$$\begin{aligned} \mathcal{X}^{(0)} &= \mathbf{m} \\ \mathcal{X}^{(i)} &= \mathbf{m} + \sqrt{n+\lambda}\,\left[\sqrt{\mathbf{P}}\right]_i \\ \mathcal{X}^{(i+n)} &= \mathbf{m} - \sqrt{n+\lambda}\,\left[\sqrt{\mathbf{P}}\right]_i, \quad i = 1, \ldots, n, \end{aligned} \tag{3.98}$$

   where $[]_i$ denotes the ith column of the matrix, and λ is a scaling parameter, which is defined in terms of algorithm parameters α and κ as follows:

$$\lambda = \alpha^2\,(n + \kappa) - n. \tag{3.99}$$

   The parameters α and κ determine the spread of the sigma points around the mean (Wan and Van der Merwe, 2001). The matrix square root denotes a matrix such that $\sqrt{\mathbf{P}}\,\sqrt{\mathbf{P}}^T = \mathbf{P}$. The sigma points are the columns of the sigma point matrix.

2. Propagate the sigma points through the non-linear function $\mathbf{g}(\cdot)$:

$$\mathcal{Y}^{(i)} = \mathbf{g}(\mathcal{X}^{(i)}), \quad i = 0, \ldots, 2n,$$

   which results in transformed sigma points $\mathcal{Y}^{(i)}$.

3. Estimates of the mean and covariance of the transformed variable can be computed from the sigma points as follows:

$$\begin{aligned} \mathrm{E}[\mathbf{g}(\mathbf{x})] &\approx \sum_{i=0}^{2n} W_i^{(m)}\,\mathcal{Y}^{(i)} \\ \mathrm{Cov}[\mathbf{g}(\mathbf{x})] &\approx \sum_{i=0}^{2n} W_i^{(c)}\,(\mathcal{Y}^{(i)} - \boldsymbol{\mu})\,(\mathcal{Y}^{(i)} - \boldsymbol{\mu})^T, \end{aligned} \tag{3.100}$$

   where the constant weights $W_i^{(m)}$ and $W_i^{(c)}$ are given as follows (Wan and Van der Merwe, 2001):

$$\begin{aligned} W_0^{(m)} &= \lambda/(n+\lambda) \\ W_0^{(c)} &= \lambda/(n+\lambda) + (1 - \alpha^2 + \beta) \\ W_i^{(m)} &= 1/\{2(n+\lambda)\}, \quad i = 1, \ldots, 2n \\ W_i^{(c)} &= 1/\{2(n+\lambda)\}, \quad i = 1, \ldots, 2n, \end{aligned} \tag{3.101}$$

   and β is an additional algorithm parameter that can be used for incorporating prior information on the (non-Gaussian) distribution of $\mathbf{x}$ (Wan and Van der Merwe, 2001).

If we apply the unscented transform to the augmented function $\tilde{\mathbf{g}}(\mathbf{x}) = (\mathbf{x}, \mathbf{g}(\mathbf{x}))$, we simply get the set of sigma points, where the sigma points $\mathcal{X}^{(i)}$ and $\mathcal{Y}^{(i)}$ have been concatenated to the same vector. Thus, also forming approximation to the joint distribution $\mathbf{x}$ and $\mathbf{g}(\mathbf{x}) + \mathbf{q}$ is straightforward and the result is:

Algorithm 3.13 (Unscented approximation of an additive transform)**.** *The unscented transform based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ *where* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_U \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_U \\ \mathbf{C}_U^T & \mathbf{S}_U \end{pmatrix} \right), \tag{3.102}$$

where the sub-matrices can be computed as follows:

1. *Form the set of* $2n+1$ *sigma points as follows:*

$$\begin{aligned} \mathcal{X}^{(0)} &= \mathbf{m} \\ \mathcal{X}^{(i)} &= \mathbf{m} + \sqrt{n+\lambda} \left[\sqrt{\mathbf{P}}\right]_i \\ \mathcal{X}^{(i+n)} &= \mathbf{m} - \sqrt{n+\lambda} \left[\sqrt{\mathbf{P}}\right]_i, \quad i = 1, \ldots, n \end{aligned} \tag{3.103}$$

   where the parameter λ *is defined in Equation* (3.99).

2. *Propagate the sigma points through the non-linear function* $\mathbf{g}(\cdot)$*:*

$$\mathcal{Y}^{(i)} = \mathbf{g}(\mathcal{X}^{(i)}), \quad i = 0, \ldots, 2n.$$

3. *The sub-matrices are then given as:*

$$\begin{aligned} \boldsymbol{\mu}_U &= \sum_{i=0}^{2n} W_i^{(m)} \, \mathcal{Y}^{(i)} \\ \mathbf{S}_U &= \sum_{i=0}^{2n} W_i^{(c)} \, (\mathcal{Y}^{(i)} - \boldsymbol{\mu}_U) \, (\mathcal{Y}^{(i)} - \boldsymbol{\mu}_U)^T + \mathbf{Q} \\ \mathbf{C}_U &= \sum_{i=0}^{2n} W_i^{(c)} \, (\mathcal{X}^{(i)} - \mathbf{m}) \, (\mathcal{Y}^{(i)} - \boldsymbol{\mu}_U)^T, \end{aligned} \tag{3.104}$$

   where the constant weights $W_i^{(m)}$ *and* $W_i^{(c)}$ *were defined in the Equation* (3.101).

The unscented transform approximation to a transformation of the form $\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{q})$ can be derived by considering the augmented random variable $\tilde{\mathbf{x}} = (\mathbf{x}, \mathbf{q})$ as the random variable in the transform. The resulting algorithm is:

Algorithm 3.14 (Unscented approximation of a non-additive transform)**.** *The unscented transform based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{q})$ *when* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_U \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_U \\ \mathbf{C}_U^T & \mathbf{S}_U \end{pmatrix} \right), \tag{3.105}$$

where the sub-matrices can be computed as follows. Let the dimensionalities of $\mathbf{x}$ *and* $\mathbf{q}$ *be* n *and* n_q*, respectively, and let* $n' = n + n_q$*.*

1. *Form the sigma points for the augmented random variable* $\tilde{\mathbf{x}} = (\mathbf{x}, \mathbf{q})$

$$\begin{aligned} \tilde{\mathcal{X}}^{(0)} &= \tilde{\mathbf{m}} \\ \tilde{\mathcal{X}}^{(i)} &= \tilde{\mathbf{m}} + \sqrt{n' + \lambda'} \left[\sqrt{\tilde{\mathbf{P}}} \right]_i \\ \tilde{\mathcal{X}}^{(i+n')} &= \tilde{\mathbf{m}} - \sqrt{n' + \lambda'} \left[\sqrt{\tilde{\mathbf{P}}} \right]_i, \quad i = 1, \ldots, n', \end{aligned} \tag{3.106}$$

where parameter λ' *is defined as in Equation* (3.99)*, but with* n *replaced by* n'*, and the augmented mean and covariance are defined by*

$$\tilde{\mathbf{m}} = \begin{pmatrix} \mathbf{m} \\ \mathbf{0} \end{pmatrix} \qquad \tilde{\mathbf{P}} = \begin{pmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{pmatrix}.$$

2. *Propagate the sigma points through the function:*

$$\tilde{\mathcal{Y}}^{(i)} = \mathbf{g}(\tilde{\mathcal{X}}^{(i),x}, \tilde{\mathcal{X}}^{(i),q}), \quad i = 0, \ldots, 2n',$$

where $\tilde{\mathcal{X}}^{(i),x}$ *and* $\tilde{\mathcal{X}}^{(i),q}$ *denote the parts of the augmented sigma point* i*, which correspond to* $\mathbf{x}$ *and* $\mathbf{q}$*, respectively.*

3. *Compute the predicted mean* $\boldsymbol{\mu}_U$*, the predicted covariance* $\mathbf{S}_U$ *and the cross-covariance* $\mathbf{C}_U$*:*

$$\begin{aligned} \boldsymbol{\mu}_U &= \sum_{i=0}^{2n'} W_i^{(m)'} \, \tilde{\mathcal{Y}}^{(i)} \\ \mathbf{S}_U &= \sum_{i=0}^{2n'} W_i^{(c)'} \, (\tilde{\mathcal{Y}}^{(i)} - \boldsymbol{\mu}_U) \, (\tilde{\mathcal{Y}}^{(i)} - \boldsymbol{\mu}_U)^T \\ \mathbf{C}_U &= \sum_{i=0}^{2n'} W_i^{(c)'} \, (\tilde{\mathcal{X}}^{(i),x} - \mathbf{m}) \, (\tilde{\mathcal{Y}}^{(i)} - \boldsymbol{\mu}_U)^T, \end{aligned}$$

where the definitions of the weights $W_i^{(m)'}$ *and* $W_i^{(c)'}$ *are the same as in Equation* (3.101)*, but with* n *replaced by* n' *and* λ *replaced by* λ'*.*

3.2.6 Unscented Kalman Filter (UKF)

The *unscented Kalman filter* (UKF) (Julier et al., 1995; Julier and Uhlmann, 2004; Wan and Van der Merwe, 2001) is an optimal filtering algorithm that utilizes the unscented transform and can be used for approximating the filtering distributions of models having the same form as with EKF and SLF, that is, models of the form (3.55) or (3.68). As EKF and SLF, UKF forms a Gaussian approximation to the filtering distribution:

$$p(\mathbf{x}_k \mid \mathbf{y}_1, \ldots, \mathbf{y}_k) \approx \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_k, \mathbf{P}_k), \tag{3.107}$$

where $\mathbf{m}_k$ and $\mathbf{P}_k$ are the mean and covariance computed by the algorithm.

Algorithm 3.15 (unscented Kalman filter I)**.** *In the additive form unscented Kalman filter (UKF) algorithm, which can be applied to additive models of the form* (3.55), *the following operations are performed at each measurement step* $k = 1, 2, 3, \ldots$*:*

1. Prediction step:

   (a) *Form the sigma points:*

$$\begin{aligned}
\mathcal{X}_{k-1}^{(0)} &= \mathbf{m}_{k-1}, \\
\mathcal{X}_{k-1}^{(i)} &= \mathbf{m}_{k-1} + \sqrt{n+\lambda} \left[\sqrt{\mathbf{P}_{k-1}}\right]_i \\
\mathcal{X}_{k-1}^{(i+n)} &= \mathbf{m}_{k-1} - \sqrt{n+\lambda} \left[\sqrt{\mathbf{P}_{k-1}}\right]_i, \quad i = 1, \ldots, n
\end{aligned} \tag{3.108}$$

   where the parameter λ *is defined in Equation* (3.99).

   (b) *Propagate the sigma points through the dynamic model:*

$$\hat{\mathcal{X}}_k^{(i)} = \mathbf{f}(\mathcal{X}_{k-1}^{(i)}), \quad i = 0, \ldots, 2n. \tag{3.109}$$

   (c) *Compute the predicted mean* $\mathbf{m}_k^-$ *and the predicted covariance* $\mathbf{P}_k^-$*:*

$$\begin{aligned}
\mathbf{m}_k^- &= \sum_{i=0}^{2n} W_i^{(m)} \, \hat{\mathcal{X}}_k^{(i)} \\
\mathbf{P}_k^- &= \sum_{i=0}^{2n} W_i^{(c)} \, (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-) \, (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)^T + \mathbf{Q}_{k-1},
\end{aligned} \tag{3.110}$$

   where the weights $W_i^{(m)}$ *and* $W_i^{(c)}$ *were defined in Equation* (3.101).

2. Update step:

   (a) *Form the sigma points:*

$$\begin{aligned}
\mathcal{X}_k^{-(0)} &= \mathbf{m}_k^-, \\
\mathcal{X}_k^{-(i)} &= \mathbf{m}_k^- + \sqrt{n+\lambda} \left[\sqrt{\mathbf{P}_k^-}\right]_i \\
\mathcal{X}_k^{-(i+n)} &= \mathbf{m}_k^- - \sqrt{n+\lambda} \left[\sqrt{\mathbf{P}_k^-}\right]_i, \quad i = 1, \ldots, n.
\end{aligned} \tag{3.111}$$

(b) *Propagate sigma points through the measurement model:*

$$\hat{\mathcal{Y}}_k^{(i)} = \mathbf{h}(\mathcal{X}_k^{-(i)}), \quad i = 0, \ldots, 2n. \tag{3.112}$$

(c) *Compute the predicted mean* $\boldsymbol{\mu}_k$*, the predicted covariance of the measurement* $\mathbf{S}_k$*, and the cross-covariance of the state and the measurement* $\mathbf{C}_k$*:*

$$\begin{aligned}
\boldsymbol{\mu}_k &= \sum_{i=0}^{2n} W_i^{(m)} \, \hat{\mathcal{Y}}_k^{(i)} \\
\mathbf{S}_k &= \sum_{i=0}^{2n} W_i^{(c)} \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k) \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T + \mathbf{R}_k \\
\mathbf{C}_k &= \sum_{i=0}^{2n} W_i^{(c)} \, (\mathcal{X}_k^{-(i)} - \mathbf{m}_k^-) \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T.
\end{aligned} \tag{3.113}$$

(d) *Compute the filter gain* $\mathbf{K}_k$*, the filtered state mean* $\mathbf{m}_k$ *and the covariance* $\mathbf{P}_k$*, conditional on the measurement* $\mathbf{y}_k$*:*

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{C}_k \, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k \, [\mathbf{y}_k - \boldsymbol{\mu}_k] \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k \, \mathbf{S}_k \, \mathbf{K}_k^T.
\end{aligned} \tag{3.114}$$

The filtering equations above can be derived in analogous manner to EKF equations, but the unscented transform based approximations are used instead of the linear approximations.

The non-additive form of UKF (Julier and Uhlmann, 2004) can be derived by augmenting the process or measurement noises with the state vector and applying UT approximation to that. Alternatively, one can first augment the state vector with process noise, then approximate the prediction step and after that do the same with measurement noise on the update step. The different algorithms and ways of doing this in practice are analyzed in article (Wu et al., 2005). However, if we directly apply the non-additive UT in the Algorithm 3.14 separately to prediction and update steps, we get the following algorithm:

Algorithm 3.16 (unscented Kalman filter II)**.** *In the augmented form unscented Kalman filter (UKF) algorithm, which can be applied to non-additive models of the form* (3.68)*, the following operations are performed at each measurement step* $k = 1, 2, 3, \ldots$*:*

1. Prediction step:

(a) *Form the sigma points for the augmented random variable* $(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})$*:*

$$\begin{aligned}
\tilde{\mathcal{X}}_{k-1}^{(0)} &= \tilde{\mathbf{m}}_{k-1}, \\
\tilde{\mathcal{X}}_{k-1}^{(i)} &= \tilde{\mathbf{m}}_{k-1} + \sqrt{n' + \lambda'} \left[\sqrt{\tilde{\mathbf{P}}_{k-1}} \right]_i \\
\tilde{\mathcal{X}}_{k-1}^{(i+n')} &= \tilde{\mathbf{m}}_{k-1} - \sqrt{n' + \lambda'} \left[\sqrt{\tilde{\mathbf{P}}_{k-1}} \right]_i, \quad i = 1, \ldots, n',
\end{aligned} \tag{3.115}$$

where

$$\tilde{\mathbf{m}}_{k-1} = \begin{pmatrix} \mathbf{m}_{k-1} \\ \mathbf{0} \end{pmatrix} \qquad \tilde{\mathbf{P}}_{k-1} = \begin{pmatrix} \mathbf{P}_{k-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_{k-1} \end{pmatrix}.$$

Here $n' = n + n_q$*, where* n *is the dimensionality of the state* $\mathbf{x}_{k-1}$ *and* n_q *is the dimensionality of the noise* $\mathbf{q}_{k-1}$*. The parameter* λ' *is defined as in Equation* (3.99)*, but with* n *replaced by* n'*.*

(b) *Propagate the sigma points through the dynamic model:*

$$\hat{\mathcal{X}}_k^{(i)} = \mathbf{f}(\tilde{\mathcal{X}}_{k-1}^{(i),x}, \tilde{\mathcal{X}}_{k-1}^{(i),q}), \quad i = 0, \ldots, 2n'. \tag{3.116}$$

where $\tilde{\mathcal{X}}_{k-1}^{(i),x}$ *denotes the first* n *components in* $\tilde{\mathcal{X}}_{k-1}^{(i)}$ *and* $\tilde{\mathcal{X}}_{k-1}^{(i),q}$ *denotes the* n_q *last components.*

(c) *Compute the predicted mean* $\mathbf{m}_k^-$ *and the predicted covariance* $\mathbf{P}_k^-$*:*

$$\begin{aligned}
\mathbf{m}_k^- &= \sum_{i=0}^{2n} W_i^{(m)'} \, \hat{\mathcal{X}}_k^{(i)} \\
\mathbf{P}_k^- &= \sum_{i=0}^{2n} W_i^{(c)'} \, (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-) \, (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)^T.
\end{aligned} \tag{3.117}$$

where the weights $W_i^{(m)'}$ *and* $W_i^{(c)'}$ *are the same as in Equation* (3.101)*, but with* n *replaced by* n' *and* λ *by* λ'*.*

2. Update step:

(a) *Form the sigma points for the augmented random variable* $(\mathbf{x}_k, \mathbf{r}_k)$*:*

$$\begin{aligned}
\tilde{\mathcal{X}}_k^{-(0)} &= \tilde{\mathbf{m}}_k^-, \\
\tilde{\mathcal{X}}_k^{-(i)} &= \tilde{\mathbf{m}}_k^- + \sqrt{n'' + \lambda''} \left[\sqrt{\tilde{\mathbf{P}}_k^-} \right]_i \\
\tilde{\mathcal{X}}_k^{-(i+n'')} &= \mathbf{m}_k^- - \sqrt{n'' + \lambda''} \left[\sqrt{\tilde{\mathbf{P}}_k^-} \right]_i, \quad i = 1, \ldots, n'',
\end{aligned} \tag{3.118}$$

where

$$\tilde{\mathbf{m}}_k^- = \begin{pmatrix} \mathbf{m}_k^- \\ \mathbf{0} \end{pmatrix} \qquad \tilde{\mathbf{P}}_k^- = \begin{pmatrix} \mathbf{P}_k^- & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_k \end{pmatrix}.$$

Here we have defined $n'' = n + n_r$, where n is the dimensionality of the state $\mathbf{x}_k$ and n_r is the dimensionality of the noise $\mathbf{r}_k$. The parameter λ'' is defined as in Equation (3.99)*, but with n replaced by n''.*

(b) *Propagate sigma points through the measurement model:*

$$\hat{\mathcal{Y}}_k^{(i)} = \mathbf{h}(\tilde{\mathcal{X}}_k^{-(i),x}, \tilde{\mathcal{X}}_k^{-(i),r}), \quad i = 0, \ldots, 2n'', \tag{3.119}$$

where $\tilde{\mathcal{X}}_k^{-(i),x}$ denotes the first n components in $\tilde{\mathcal{X}}_k^{-(i)}$ and $\tilde{\mathcal{X}}_k^{-(i),r}$ denotes the n_r last components.

(c) *Compute the predicted mean $\boldsymbol{\mu}_k$, the predicted covariance of the measurement $\mathbf{S}_k$, and the cross-covariance of the state and the measurement $\mathbf{C}_k$:*

$$\begin{aligned}
\boldsymbol{\mu}_k &= \sum_{i=0}^{2n''} W_i^{(m)''}\, \hat{\mathcal{Y}}_k^{(i)} \\
\mathbf{S}_k &= \sum_{i=0}^{2n''} W_{i-1}^{(c)''}\, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)\, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T \\
\mathbf{C}_k &= \sum_{i=0}^{2n''} W_i^{(c)''}\, (\mathcal{X}_k^{-(i),x} - \mathbf{m}_k^-)\, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T,
\end{aligned} \tag{3.120}$$

where the weights $W_i^{(m)''}$ and $W_i^{(c)''}$ are the same as in Equation (3.101)*, but with n replaced by n'' and λ by λ''.*

(d) *Compute the filter gain $\mathbf{K}_k$ and the filtered state mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$, conditional to the measurement $\mathbf{y}_k$:*

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{C}_k\, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, [\mathbf{y}_k - \boldsymbol{\mu}_k] \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T.
\end{aligned} \tag{3.121}$$

The advantage of the UKF over EKF is that UKF is not based on a local linear approximation, but uses a bit further points in approximating the non-linearity. As discussed in Julier and Uhlmann (2004) the unscented transform is able to capture the higher order moments caused by the non-linear transform better than the Taylor series based approximations. However, an important point to note is that although the mean estimate of UT is exact for polynomials up to order 3, the covariance computation is only exact for polynomials up to the first order (as, e.g., in SLF). In UT, the dynamic and model functions are also not required to be formally differentiable nor do their Jacobian matrices need to be computed. The advantage of UKF over SLF is that in UKF there is no need to compute any expected values in closed form, only evaluations of the dynamic and measurement models are needed. However, the accuracy of UKF cannot be expected to be as good as that of SLF, because SLF uses a larger area in the approximation, whereas UKF only selects a fixed number of points in the area. The disadvantage over EKF is that UKF often requires slightly more computational operations than EKF.

The UKF can be interpreted to belong to a wider class of filters called sigma-point filters (Van der Merwe and Wan, 2003), which also includes other types of filters such as central differences Kalman filter (CDKF), Gauss-Hermite Kalman filter (GHKF) and a few others (Ito and Xiong, 2000; Wu et al., 2006; Nørgaard et al., 2000; Arasaratnam and Haykin, 2009). The classification to sigma-point methods by Van der Merwe and Wan (2003) is based on interpreting the methods as special cases of (weighted) statistical linear regression (Lefebvre et al., 2002). As discussed in (Van der Merwe and Wan, 2003), statistical linearization is closely related to sigma-point approximations, because they both are related to statistical linear regression. However, it is important to note that the statistical linear regression (Lefebvre et al., 2002) which is the basis of sigma-point framework (Van der Merwe and Wan, 2003) is not exactly equivalent to statistical linearization (Gelb, 1974) as sometimes is claimed. The statistical linear regression can be considered as a discrete approximation to statistical linearization.

3.2.7 Fourier-Hermite Series Expansions

In this section, we show how Fourier-Hermite series can be used for approximating non-linear transformations of Gaussian random variables and how this approach can be seen as a generalization of statistical linearization. The presentation here is based on the article by Sarmavuori and Särkkä (2012).

In Section 3.2.3 we formed the statistical linearization based approximation to the transformation

$$\begin{aligned} \mathbf{x} &\sim \mathrm{N}(\mathbf{m}, \mathbf{P}) \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}) \end{aligned} \tag{3.122}$$

by postulating the approximation

$$\mathbf{g}(\mathbf{x}) \approx \mathbf{b} + \mathbf{A}\,\delta\mathbf{x}, \tag{3.123}$$

and by finding the optimal matrix $\mathbf{A}$ and vector $\mathbf{b}$ by minimizing $\mathrm{E}[||\mathbf{g}(\mathbf{x}) - \mathbf{b} - \mathbf{A}\,\delta\,\mathbf{x}||^2]$, where $\delta\mathbf{x} = \mathbf{x} - \mathbf{m}$. We could now attempt to generalize this such that instead of the linear approximation, we use a pth order polynomial approximation

$$\mathbf{g}(\mathbf{x}) \approx \mathbf{b} + \mathbf{A}\,\delta\mathbf{x} + \delta\mathbf{x}^T\,\mathbf{C}\,\delta\mathbf{x} + \ldots \tag{3.124}$$

By expanding the expression and setting derivatives to zero we could determine the optimal polynomial coefficients. This indeed is possible, but with higher order polynomials this approach quickly becomes tedious. Fortunately, we can formulate the approximation more conveniently in terms of Hilbert space theory by defining an inner product for scalar functions g and f as follows:

$$\begin{aligned}\langle f, g\rangle &= \int f(\mathbf{x})\, g(\mathbf{x})\ \mathrm{N}(\mathbf{x}\,|\,\mathbf{m},\mathbf{P})\,\mathrm{d}\mathbf{x}\\ &= \mathrm{E}[f(\mathbf{x})\, g(\mathbf{x})],\end{aligned} \tag{3.125}$$

where $\mathbf{x} \sim \mathrm{N}(\mathbf{m},\mathbf{P})$. We can now form a Hilbert space of functions by defining a norm as

$$||g||^2 = \langle g, g\rangle. \tag{3.126}$$

Hilbert space theory now tells that there exists an orthogonal polynomial basis of the corresponding Hilbert space. It turns out that these polynomial basis functions are scaled multivariate Hermite polynomials, which we here define as[3]

$$H_{[a_1,\ldots,a_p]}(\mathbf{x};\mathbf{m},\mathbf{P}) = H_{[a_1,\ldots,a_p]}(\mathbf{L}^{-1}\,(\mathbf{x}-\mathbf{m})), \tag{3.127}$$

where $\mathbf{L}$ is a matrix such that $\mathbf{P} = \mathbf{L}\,\mathbf{L}^T$ and

$$H_{[a_1,\ldots,a_p]}(\mathbf{x}) = (-1)^p\ \exp(||\mathbf{x}||^2/2)\frac{\partial^n}{\partial x_{a_1}\cdots\partial x_{a_p}}\exp(-||\mathbf{x}||^2/2). \tag{3.128}$$

We can now expand an arbitrary vector function $\mathbf{g}(\mathbf{x})$ with $\langle g_i, g_i\rangle < \infty$ into a Fourier-Hermite series as follows:

$$\mathbf{g}(\mathbf{x}) = \sum_{k=0}^{\infty}\ \sum_{a_1,\ldots,a_k=1}^{n} \frac{1}{k!}\ \mathrm{E}[\mathbf{g}(\mathbf{x})\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P})]\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P}). \tag{3.129}$$

Notice that because $H_{[0,\ldots,0]}(\mathbf{x};\mathbf{m},\mathbf{P}) = 1$, the zeroth order term in the series is just the expectation $\mathrm{E}[\mathbf{g}(\mathbf{x})]$. By using the orthogonality of the basis functions we get that the sum of expectations of outer products is

$$\begin{aligned}\mathrm{E}[\mathbf{g}(\mathbf{x})\,\mathbf{g}^T(\mathbf{x})] = \sum_{k=0}^{\infty}\ \sum_{a_1,\ldots,a_k=1}^{n} \frac{1}{k!}\ &\mathrm{E}[\mathbf{g}(\mathbf{x})\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P})]\\ &\times \mathrm{E}[\mathbf{g}^T(\mathbf{x})\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P})].\end{aligned} \tag{3.130}$$

By leaving out the zeroth order term we get the following exact representation for the covariance of $\mathbf{g}(x)$:

$$\begin{aligned}\mathrm{Cov}[\mathbf{g}(\mathbf{x})] = \sum_{k=1}^{\infty}\ \sum_{a_1,\ldots,a_k=1}^{n} \frac{1}{k!}\ &\mathrm{E}[\mathbf{g}(\mathbf{x})\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P})]\\ &\times \mathrm{E}[\mathbf{g}^T(\mathbf{x})\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P})].\end{aligned} \tag{3.131}$$

[3]Note that this definition used here as well as in Kuznetsov et al. (1960); Sarmavuori and Särkkä (2012) differs from the "natural" definition $H_{(p_1,\ldots,p_n)}(\mathbf{x};\mathbf{m},\mathbf{P}) = H_{(p_1,\ldots,p_n)}(\mathbf{L}^{-1}(\mathbf{x}-\mathbf{m}))$, with $H_{(p_1,\ldots,p_n)}(\mathbf{x}) = H_{p_1}(x_1)\cdots H_{p_n}(x_n)$, where H_{p_j} are the univariate Hermite polynomials, because this way the notation remains clearer (Sarmavuori and Särkkä, 2012). The different definitions are of course equivalent.

From Hilbert space theory we know that the best pth polynomial approximation to $\mathbf{g}(\mathbf{x})$ with respect to $||\cdot||^2$, that is, the polynomial expansion (3.124) is given by the orthogonal projection on the Hermite polynomials up to order p. Thus the optimal pth order polynomial approximation is given by truncating the series (3.129) at order p.

We could now consider computing the coefficients of the series with some numerical method and then pick the zeroth order term for the mean and compute an approximation to the covariance by the truncated series above. However, this does not make much sense, because with the same numerical method we could equivalently directly compute the covariance as well. Fortunately, the Fourier-Hermite series coefficients can be computed in an alternative way because of the following result (Sarmavuori and Särkkä, 2012):

$$\mathrm{E}[\mathbf{g}(\mathbf{x})\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P})] = \sum_{b_1,\ldots,b_k=1}^{n} \mathrm{E}\left[\frac{\partial^k \mathbf{g}(\mathbf{x})}{\partial x_{b_1}\cdots\partial x_{b_k}}\right] \prod_{m=1}^{k} L_{b_m,a_m}, \tag{3.132}$$

which is a generalization of the derivative version of the statistical linearization discussed in Section 3.2.3. Thus we can express the Fourier-Hermite series as follows:

$$\mathbf{g}(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\substack{a_1,\ldots,a_k=1\\ b_1,\ldots,b_k=1}}^{n} \frac{1}{k!}\,\mathrm{E}\left[\frac{\partial^k \mathbf{g}(\mathbf{x})}{\partial x_{b_1}\cdots\partial x_{b_k}}\right] \prod_{m=1}^{k} L_{b_m,a_m}\, H_{[a_1,\ldots,a_k]}(\mathbf{x};\mathbf{m},\mathbf{P}), \tag{3.133}$$

and the covariance as:

$$\begin{aligned}\mathrm{Cov}[\mathbf{g}(\mathbf{x})\,\mathbf{g}^T(\mathbf{x})] = \sum_{k=1}^{\infty} \sum_{\substack{a_1,\ldots,a_k=1\\ b_1,\ldots,b_k=1}}^{n} \frac{1}{k!}\,\mathrm{E}\left[\frac{\partial^k \mathbf{g}(\mathbf{x})}{\partial x_{b_1}\cdots\partial x_{b_k}}\right] \prod_{m=1}^{k} P_{b_m,a_m} \\ \times\,\mathrm{E}\left[\frac{\partial^k \mathbf{g}(\mathbf{x})}{\partial x_{a_1}\cdots\partial x_{a_k}}\right]^T.\end{aligned} \tag{3.134}$$

It turns out that we do not even need to compute the expectations of the derivatives, because they can be evaluated as follows (Sarmavuori and Särkkä, 2012):

- Assume that we can compute the following expectation in closed form:

$$\hat{\mathbf{g}}(\mathbf{m},\mathbf{P}) = \mathrm{E}[\mathbf{g}(\mathbf{x})] = \int \mathbf{g}(\mathbf{x})\; \mathrm{N}(\mathbf{x}\,|\,\mathbf{m},\mathbf{P})\,\mathrm{d}\mathbf{x}, \tag{3.135}$$

  for an arbitrary $\mathbf{m}$ and $\mathbf{P}$.

- Then we have

$$\mathrm{E}\left[\frac{\partial^k \mathbf{g}(\mathbf{x})}{\partial x_{b_1},\ldots,\partial x_{b_k}}\right] = \frac{\partial^k \hat{\mathbf{g}}(\mathbf{m},\mathbf{P})}{\partial m_{b_1}\cdots\partial m_{b_k}} \tag{3.136}$$

In forming the approximation to the joint mean and covariance of the augmented function $\tilde{\mathbf{g}}(\mathbf{x}) = (\mathbf{x}, \mathbf{g}(\mathbf{x}))$ we also need the following term, which can also be computed using the above result:

$$\mathrm{E}[(\mathbf{x}-\mathbf{m})\,(\mathbf{g}(\mathbf{x})-\mathrm{E}[\mathbf{g}(\mathbf{x})])^T] = \mathbf{P}\;\mathrm{E}[\mathbf{G}_{\mathbf{x}}(\mathbf{x})]^T = \mathbf{P}\,\hat{\mathbf{G}}^T, \tag{3.137}$$

where $\hat{G}_{ij} = \partial \hat{g}_i / \partial m_j$. We get the following algorithm:

Algorithm 3.17 (Fourier-Hermite series approximation of an additive transform)**.** *The Fourier-Hermite series based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ *where* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix}\mathbf{m}\\ \boldsymbol{\mu}_F\end{pmatrix}, \begin{pmatrix}\mathbf{P} & \mathbf{C}_F\\ \mathbf{C}_F^T & \mathbf{S}_F\end{pmatrix}\right), \tag{3.138}$$

where

$$\begin{aligned}
\boldsymbol{\mu}_F &= \hat{\mathbf{g}}(\mathbf{m}, \mathbf{P})\\
\mathbf{S}_F &= \mathbf{Q} + \underbrace{\sum_{ij} \hat{\mathbf{g}}_i^{(1)}(\mathbf{m}, \mathbf{P})\, P_{ij}\, [\hat{\mathbf{g}}_j^{(1)}(\mathbf{m}, \mathbf{P})]^T}_{\hat{\mathbf{G}}\,\mathbf{P}\,\hat{\mathbf{G}}^T}\\
&\quad + \frac{1}{2!}\sum_{ijuv} \hat{\mathbf{g}}_{iu}^{(2)}(\mathbf{m}, \mathbf{P})\, P_{ij}\, P_{uv}\, [\hat{\mathbf{g}}_{jv}^{(2)}(\mathbf{m}, \mathbf{P})]^T\\
&\quad + \frac{1}{3!}\sum_{ijuvpq} \hat{\mathbf{g}}_{iup}^{(3)}(\mathbf{m}, \mathbf{P})\, P_{ij}\, P_{uv}\, P_{pq}\, [\hat{\mathbf{g}}_{jvq}^{(3)}(\mathbf{m}, \mathbf{P})]^T\\
&\quad + \ldots\\
\mathbf{C}_F &= \mathbf{P}\,\hat{\mathbf{G}}^T,
\end{aligned} \tag{3.139}$$

and we have defined

$$\begin{aligned}
\hat{\mathbf{g}}(\mathbf{m}, \mathbf{P}) &= \int \mathbf{g}(\mathbf{x})\;\mathrm{N}(\mathbf{x} \mid \mathbf{m}, \mathbf{P})\,\mathrm{d}\mathbf{x}\\
\hat{\mathbf{g}}_{b_1,\ldots,b_k}^{(k)}(\mathbf{m}, \mathbf{P}) &= \frac{\partial^k \hat{\mathbf{g}}(\mathbf{m}, \mathbf{P})}{\partial m_{b_1} \cdots \partial m_{b_k}}
\end{aligned} \tag{3.140}$$

and the matrix $\hat{\mathbf{G}}$ *is defined as*

$$\hat{G}_{ij} = [\hat{g}_j^{(1)}(\mathbf{m}, \mathbf{P})]_i. \tag{3.141}$$

The mean and cross-covariance in the above approximation are always exact (assuming that we can compute them exactly) and when the series is truncated at order p, the covariance is accurate for polynomials up to order p. The first order approximation is equivalent to the statistically linearized approximation. The approximation to the non-additive transform can be obtained analogously.

3.2.8 Fourier-Hermite Kalman Filter

In this section we present the Fourier-Hermite Kalman filter (FHKF, Sarmavuori and Särkkä, 2012), which is based on the Fourier-Hermite series expansion in the previous section and is a generalization of the statistically linearized filter. To implement the algorithm, we need closed form expressions or good approximations to the following expressions with arbitrary $\mathbf{m}$ and $\mathbf{P}$:

$$\begin{aligned} \hat{\mathbf{f}}(\mathbf{m},\mathbf{P}) &= \int \mathbf{f}(\mathbf{x})\; \mathrm{N}(\mathbf{x}\,|\,\mathbf{m},\mathbf{P})\,\mathrm{d}\mathbf{x} \\ \hat{\mathbf{f}}^{(k)}_{b_1,\ldots,b_k}(\mathbf{m},\mathbf{P}) &= \frac{\partial^k \hat{\mathbf{f}}(\mathbf{m},\mathbf{P})}{\partial m_{b_1}\cdots\partial m_{b_k}}, \end{aligned} \tag{3.142}$$

as well as to the following expressions:

$$\begin{aligned} \hat{\mathbf{h}}(\mathbf{m},\mathbf{P}) &= \int \mathbf{h}(\mathbf{x})\; \mathrm{N}(\mathbf{x}\,|\,\mathbf{m},\mathbf{P})\,\mathrm{d}\mathbf{x} \\ \hat{\mathbf{h}}^{(k)}_{b_1,\ldots,b_k}(\mathbf{m},\mathbf{P}) &= \frac{\partial^k \hat{\mathbf{h}}(\mathbf{m},\mathbf{P})}{\partial m_{b_1}\cdots\partial m_{b_k}}. \end{aligned} \tag{3.143}$$

The filter is the following:

Algorithm 3.18 (Fourier-Hermite Kalman filter)**.** *The prediction and update steps of the additive noise Fourier-Hermite Kalman filter (FHKF) are:*

- *Prediction:*

$$\begin{aligned} \mathbf{m}_k^- &= \hat{\mathbf{f}}(\mathbf{m}_{k-1},\mathbf{P}_{k-1}) \\ \mathbf{P}_k^- &= \mathbf{Q}_{k-1} + \sum_{ij} \hat{\mathbf{f}}^{(1)}_i(\mathbf{m}_{k-1},\mathbf{P}_{k-1})\,[P_{k-1}]_{ij}\,[\hat{\mathbf{f}}^{(1)}_j(\mathbf{m}_{k-1},\mathbf{P}_{k-1})]^T \\ &\quad + \frac{1}{2!}\sum_{ijuv} \hat{\mathbf{f}}^{(2)}_{iu}(\mathbf{m}_{k-1},\mathbf{P}_{k-1})\,[P_{k-1}]_{ij}\,[P_{k-1}]_{uv}\,[\hat{\mathbf{f}}^{(2)}_{jv}(\mathbf{m}_{k-1},\mathbf{P}_{k-1})]^T \\ &\quad + \frac{1}{3!}\sum_{ijuvpq} \hat{\mathbf{f}}^{(3)}_{iup}(\mathbf{m}_{k-1},\mathbf{P}_{k-1})\,[P_{k-1}]_{ij}\,[P_{k-1}]_{uv}\,[P_{k-1}]_{pq} \\ &\qquad\qquad \times [\hat{\mathbf{f}}^{(3)}_{jvq}(\mathbf{m}_{k-1},\mathbf{P}_{k-1})]^T \\ &\quad + \ldots \end{aligned} \tag{3.144}$$

- *Update:*

$$
\begin{aligned}
\mathbf{v}_k &= \mathbf{y}_k - \hat{\mathbf{h}}(\mathbf{m}_k^-, \mathbf{P}_k^-) \\
\mathbf{S}_k &= \mathbf{R}_k \sum_{ij} \hat{\mathbf{h}}_i^{(1)}(\mathbf{m}_k^-, \mathbf{P}_k^-)\, [P_k^-]_{ij}\, [\hat{\mathbf{h}}_j^{(1)}(\mathbf{m}_k^-, \mathbf{P}_k^-)]^T \\
&\quad + \frac{1}{2!} \sum_{ijuv} \hat{\mathbf{h}}_{iu}^{(2)}(\mathbf{m}_k^-, \mathbf{P}_k^-)\, [P_k^-]_{ij}\, [P_k^-]_{uv}\, [\hat{\mathbf{h}}_{jv}^{(2)}(\mathbf{m}_k^-, \mathbf{P}_k^-)]^T \\
&\quad + \frac{1}{3!} \sum_{ijuvpq} \hat{\mathbf{h}}_{iup}^{(3)}(\mathbf{m}_k^-, \mathbf{P}_k^-)\, [P_k^-]_{ij}\, [P_k^-]_{uv}\, [P_k^-]_{pq} \\
&\qquad\qquad \times [\hat{\mathbf{h}}_{jvq}^{(3)}(\mathbf{m}_k^-, \mathbf{P}_k^-)]^T \\
&\quad + \ldots
\end{aligned}
\tag{3.145}
$$

$$
\begin{aligned}
\mathbf{K}_k &= \mathbf{P}_k^-\, \hat{\mathbf{H}}^T\, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, \mathbf{v}_k \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T,
\end{aligned}
$$

where the matrix $\hat{\mathbf{H}}$ *is defined as*

$$
\hat{H}_{ij} = [\hat{h}_j^{(1)}(\mathbf{m}_k^-, \mathbf{P}_k^-)]_i. \tag{3.146}
$$

3.3 Gaussian Filtering

Quite soon after the unscented Kalman filter (UKF) was published, Ito and Xiong (2000) pointed out that UKF can be considered as a special case of so called Gaussian filters, where the non-linear filtering problem is solved using Gaussian assumed density approximations. The generalized framework also enables the usage of various powerful Gaussian quadrature and cubature integration methods (Wu et al., 2006; Arasaratnam and Haykin, 2009). The series expansion based filters presented in the previous sections can also be seen as approximations to the general Gaussian filter. In this section we present the Gaussian filtering framework and show how the Gauss-Hermite Kalman filter (GHKF) and the cubature Kalman filter (CKF) can be derived as its approximations. We also show how UKF can be seen as a generalization of CKF.

3.3.1 Gaussian Moment Matching

One way to unify various Gaussian approximation based approaches is to think all of them as approximations to Gaussian integrals of the form:

$$
\int \mathbf{g}(\mathbf{x})\ \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, \mathrm{d}\mathbf{x}.
$$

If we can compute these, a straight-forward way to form the Gaussian approximation for $(\mathbf{x}, \mathbf{y})$ is to simply match the moments of the distributions, which gives the following algorithm:

Algorithm 3.19 (Gaussian moment matching of an additive transform)**.** *The moment matching based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}) + \mathbf{q}$ *where* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_M \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_M \\ \mathbf{C}_M^T & \mathbf{S}_M \end{pmatrix} \right), \tag{3.147}$$

where

$$\begin{aligned} \boldsymbol{\mu}_M &= \int \mathbf{g}(\mathbf{x}) \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \, \mathrm{d}\mathbf{x} \\ \mathbf{S}_M &= \int (\mathbf{g}(\mathbf{x}) - \boldsymbol{\mu}_M) \, (\mathbf{g}(\mathbf{x}) - \boldsymbol{\mu}_M)^T \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \, \mathrm{d}\mathbf{x} + \mathbf{Q} \\ \mathbf{C}_M &= \int (\mathbf{x} - \mathbf{m}) \, (\mathbf{g}(\mathbf{x}) - \boldsymbol{\mu}_M)^T \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \, \mathrm{d}\mathbf{x}. \end{aligned} \tag{3.148}$$

It is now easy to check by substituting the approximation $\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{m}) + \mathbf{G}_{\mathbf{x}}(\mathbf{m}) \, (\mathbf{x} - \mathbf{m})$ to the above expression that in the linear case the integrals indeed reduce to the linear approximations in the Algorithm 3.2. And the same applies to statistical linearization. However, many other approximations can also be interpreted as such approximations as is discussed in the next section.

The non-additive version of the transform is the following:

Algorithm 3.20 (Gaussian moment matching of a non-additive transform)**.** *The moment matching based Gaussian approximation to the joint distribution of* $\mathbf{x}$ *and the transformed random variable* $\mathbf{y} = \mathbf{g}(\mathbf{x}, \mathbf{q})$ *where* $\mathbf{x} \sim \mathrm{N}(\mathbf{m}, \mathbf{P})$ *and* $\mathbf{q} \sim \mathrm{N}(\mathbf{0}, \mathbf{Q})$ *is given as*

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu}_M \end{pmatrix}, \begin{pmatrix} \mathbf{P} & \mathbf{C}_M \\ \mathbf{C}_M^T & \mathbf{S}_M \end{pmatrix} \right), \tag{3.149}$$

where

$$\begin{aligned} \boldsymbol{\mu}_M &= \int \mathbf{g}(\mathbf{x}, \mathbf{q}) \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \; \mathrm{N}(\mathbf{q} \,|\, \mathbf{0}, \mathbf{Q}) \, \mathrm{d}\mathbf{x} \, \mathrm{d}\mathbf{q} \\ \mathbf{S}_M &= \int (\mathbf{g}(\mathbf{x}, \mathbf{q}) - \boldsymbol{\mu}_M) \, (\mathbf{g}(\mathbf{x}, \mathbf{q}) - \boldsymbol{\mu}_M)^T \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \; \mathrm{N}(\mathbf{q} \,|\, \mathbf{0}, \mathbf{Q}) \, \mathrm{d}\mathbf{x} \, \mathrm{d}\mathbf{q} \\ \mathbf{C}_M &= \int (\mathbf{x} - \mathbf{m}) \, (\mathbf{g}(\mathbf{x}, \mathbf{q}) - \boldsymbol{\mu}_M)^T \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \; \mathrm{N}(\mathbf{q} \,|\, \mathbf{0}, \mathbf{Q}) \, \mathrm{d}\mathbf{x} \, \mathrm{d}\mathbf{q}. \end{aligned} \tag{3.150}$$

3.3.2 Gaussian Filter

If we replace the linear approximations in EKF with the moment matching approximations in the previous section, we get the following *Gaussian assumed density filter* (ADF) which is also called *Gaussian filter* (Maybeck, 1982a; Ito and Xiong, 2000; Wu et al., 2006):

Algorithm 3.21 (Gaussian filter I)**.** *The prediction and update steps of the additive noise Gaussian (Kalman) filter are:*

- *Prediction:*

$$\begin{aligned}
\mathbf{m}_k^- &= \int \mathbf{f}(\mathbf{x}_{k-1})\ \mathrm{N}(\mathbf{x}_{k-1} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1})\, \mathrm{d}\mathbf{x}_{k-1} \\
\mathbf{P}_k^- &= \int (\mathbf{f}(\mathbf{x}_{k-1}) - \mathbf{m}_k^-)\, (\mathbf{f}(\mathbf{x}_{k-1}) - \mathbf{m}_k^-)^T \\
&\qquad \times \mathrm{N}(\mathbf{x}_{k-1} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1})\, \mathrm{d}\mathbf{x}_{k-1} + \mathbf{Q}_{k-1}.
\end{aligned} \tag{3.151}$$

- *Update:*

$$\begin{aligned}
\boldsymbol{\mu}_k &= \int \mathbf{h}(\mathbf{x}_k)\ \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-)\, \mathrm{d}\mathbf{x}_k \\
\mathbf{S}_k &= \int (\mathbf{h}(\mathbf{x}_k) - \boldsymbol{\mu}_k)\, (\mathbf{h}(\mathbf{x}_k) - \boldsymbol{\mu}_k)^T\ \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-)\, \mathrm{d}\mathbf{x}_k + \mathbf{R}_k \\
\mathbf{C}_k &= \int (\mathbf{x}_k - \mathbf{m}^-)\, (\mathbf{h}(\mathbf{x}_k) - \boldsymbol{\mu}_k)^T\ \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-)\, \mathrm{d}\mathbf{x}_k \\
\mathbf{K}_k &= \mathbf{C}_k\, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, (\mathbf{y}_k - \boldsymbol{\mu}_k) \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T.
\end{aligned} \tag{3.152}$$

The advantage of the moment matching formulation is that it enables usage of many well known numerical integration methods such as Gauss-Hermite quadratures, cubature rules and central difference based methods (Ito and Xiong, 2000; Wu et al., 2006; Nørgaard et al., 2000; Arasaratnam and Haykin, 2009). The unscented transformation can also be interpreted as an approximation to these integrals (Wu et al., 2006).

One interesting way to approximate the integrals is to use the Bayes-Hermite quadrature (O'Hagan, 1991), which is based of fitting a Gaussian process regression model to the non-linear functions on finite set of training points. This approach is used in the Gaussian process filter of Deisenroth et al. (2009). It is also possible to approximate the integrals by Monte Carlo integration, which is the approach used in Monte Carlo Kalman Filter (MCKF). That idea can also be extended to non-Gaussian measurement models (Kotecha and Djuric, 2003).

The Gaussian filter can be extended to non-additive noise models as follows:

Algorithm 3.22 (Gaussian filter II)**.** *The prediction and update steps of the non-additive noise Gaussian (Kalman) filter are:*

- *Prediction:*

$$\begin{aligned}
\mathbf{m}_k^- &= \int \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) \\
&\qquad \times \mathrm{N}(\mathbf{x}_{k-1} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1}) \; \mathrm{N}(\mathbf{q}_{k-1} \,|\, \mathbf{0}, \mathbf{Q}_{k-1}) \, \mathrm{d}\mathbf{x}_{k-1} \, \mathrm{d}\mathbf{q}_{k-1} \\
\mathbf{P}_k^- &= \int (\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) - \mathbf{m}_k^-) \, (\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) - \mathbf{m}_k^-)^T \\
&\qquad \times \mathrm{N}(\mathbf{x}_{k-1} \,|\, \mathbf{m}_{k-1}, \mathbf{P}_{k-1}) \; \mathrm{N}(\mathbf{q}_{k-1} \,|\, \mathbf{0}, \mathbf{Q}_{k-1}) \, \mathrm{d}\mathbf{x}_{k-1} \, \mathrm{d}\mathbf{q}_{k-1}.
\end{aligned} \tag{3.153}$$

- *Update:*

$$\begin{aligned}
\boldsymbol{\mu}_k &= \int \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k) \\
&\qquad \times \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-) \; \mathrm{N}(\mathbf{r}_k \,|\, \mathbf{0}, \mathbf{R}_k) \, \mathrm{d}\mathbf{x}_k \, \mathrm{d}\mathbf{r}_k \\
\mathbf{S}_k &= \int (\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k) - \boldsymbol{\mu}_k) \, (\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k) - \boldsymbol{\mu}_k)^T \\
&\qquad \times \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-) \; \mathrm{N}(\mathbf{r}_k \,|\, \mathbf{0}, \mathbf{R}_k) \, \mathrm{d}\mathbf{x}_k \, \mathrm{d}\mathbf{r}_k \\
\mathbf{C}_k &= \int (\mathbf{x}_k - \mathbf{m}^-) \, (\mathbf{h}(\mathbf{x}_k, \mathbf{r}_k) - \boldsymbol{\mu}_k)^T \\
&\qquad \times \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^-, \mathbf{P}_k^-) \; \mathrm{N}(\mathbf{r}_k \,|\, \mathbf{0}, \mathbf{R}_k) \, \mathrm{d}\mathbf{x}_k \, \mathrm{d}\mathbf{r}_k \\
\mathbf{K}_k &= \mathbf{C}_k \, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k \, (\mathbf{y}_k - \boldsymbol{\mu}_k) \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k \, \mathbf{S}_k \, \mathbf{K}_k^T.
\end{aligned} \tag{3.154}$$

3.3.3 Gauss-Hermite Integration

In the Gaussian filter (and later in smoother) we are interested in approximating Gaussian integrals of the form

$$\begin{aligned}
&\int \mathbf{g}(\mathbf{x}) \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \, \mathrm{d}\mathbf{x} \\
&\qquad = \frac{1}{(2\,\pi)^{n/2} \, |\mathbf{P}|^{1/2}} \int \mathbf{g}(\mathbf{x}) \, \exp\left(-\frac{1}{2} (\mathbf{x} - \mathbf{m})^T \, \mathbf{P}^{-1} \, (\mathbf{x} - \mathbf{m}) \right) \, \mathrm{d}\mathbf{x},
\end{aligned} \tag{3.155}$$

where $\mathbf{g}(\mathbf{x})$ is an arbitrary function. In this section, we shall derive a Gauss-Hermite based numerical cubature[4] algorithm for computing such integrals. The

[4]As one-dimensional integrals are *quadratures*, multidimensional integrals have been traditionally called *cubatures*.

algorithm is based on direct generalization of the one-dimensional Gauss-Hermite rule into multiple dimensions by taking Cartesian product of one-dimensional quadratures. The disadvantage of the method is that the required number of evaluation points is exponential with respect to the number of dimensions.

In its basic form, one-dimensional Gauss-Hermite quadrature integration refers to the special case of Gaussian quadratures with unit Gaussian weight function $w(x) = \mathrm{N}(x \,|\, 0, 1)$, that is, to approximations of the form

$$\int_{-\infty}^{\infty} g(x)\ \mathrm{N}(x \,|\, 0, 1)\,\mathrm{d}x \approx \sum_i W^{(i)} g(x^{(i)}), \tag{3.156}$$

where $W^{(i)}, i = 1, \ldots, p$ are the weights and $x^{(i)}$ are the evaluation points or abscissas — also sometimes called sigma points. Note that the quadrature is often defined for the weight function $\exp(-x^2)$, but here we shall use the “probabilists’ definition” above. The two versions of the quadrature are related by simple scaling of variables.

Obviously, there is an infinite number of possible ways to select the weights and evaluation points. In Gauss-Hermite integration, as in all Gaussian quadratures, the weights and sigma points are chosen such that with polynomial integrand the approximation becomes exact. It turns out that the polynomial order with given number of points is maximized is we choose the sigma points to be roots of Hermite polynomials. When using pth order Hermite polynomial $H_p(x)$, the rule will be exact for polynomials up to order $2p - 1$. The required weights can be computed in closed form (see below).

The Hermite polynomial of order p is defined as (these are so called “probabilists’ Hermite polynomials”):

$$H_p(x) = (-1)^p \exp(x^2/2)\,\frac{d^p}{dx^p} \exp(-x^2/2). \tag{3.157}$$

The first few Hermite polynomials are:

$$\begin{aligned} H_0(x) &= 1 \\ H_1(x) &= x \\ H_2(x) &= x^2 - 1 \\ H_3(x) &= x^3 - 3x \\ H_4(x) &= x^4 - 6\,x^2 + 3, \end{aligned} \tag{3.158}$$

and further polynomials can be found from the recursion

$$H_{p+1}(x) = x\,H_p(x) - p\,H_{p-1}(x). \tag{3.159}$$

Using the same weights and sigma points, integrals over non-unit Gaussian weights functions $\mathrm{N}(x \,|\, m, P)$ can be evaluated using a simple change of integration variable:

$$\int_{-\infty}^{\infty} g(x)\ \mathrm{N}(x \,|\, m, P)\,\mathrm{d}x = \int_{-\infty}^{\infty} g(P^{1/2}\,\xi + m)\ \mathrm{N}(\xi \,|\, 0, 1)\,\mathrm{d}\xi \tag{3.160}$$

The Gauss-Hermite integration can be written as the following algorithm:

Algorithm 3.23 (Gauss-Hermite quadrature)**.** *The pth order Gauss-Hermite approximation to the 1-dimensional integral*

$$\int_{-\infty}^{\infty} g(x)\; \mathrm{N}(x \,|\, m, P)\, \mathrm{d}x \tag{3.161}$$

can be computed as follows:

1. *Compute the unit sigma points as the roots $\xi^{(i)}, i = 1, \ldots, p$ of Hermite polynomial $H_p(x)$. Note that we do not need to form the polynomial and them compute its roots, but instead it is numerically more stable to compute the roots as eigenvalues of a suitable tridiagonal matrix (Golub and Welsch, 1969).*
2. *Compute the weights as*
$$W^{(i)} = \frac{p!}{p^2\, [H_{p-1}(\xi^{(i)})]^2}. \tag{3.162}$$
3. *Approximate the integral as*
$$\int_{-\infty}^{\infty} g(x)\; \mathrm{N}(x \,|\, m, P)\, \mathrm{d}x \approx \sum_{i=1}^{p} W^{(i)} g(P^{1/2}\, \xi^{(i)} + m). \tag{3.163}$$

By generalizing the change of variables idea, we can form approximations to multidimensional integrals of the form (3.155). First let $\mathbf{P} = \sqrt{\mathbf{P}}\,\sqrt{\mathbf{P}}^T$, where $\sqrt{\mathbf{P}}$ is the Cholesky factor of the covariance matrix $\mathbf{P}$ or some other similar square root of the covariance matrix. If we define new integration variables $\boldsymbol{\xi}$ by

$$\mathbf{x} = \mathbf{m} + \sqrt{\mathbf{P}}\, \boldsymbol{\xi}, \tag{3.164}$$

we get

$$\int \mathbf{g}(\mathbf{x})\; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})\, \mathrm{d}\mathbf{x} = \int \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\, \boldsymbol{\xi})\; \mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I})\, \mathrm{d}\boldsymbol{\xi}. \tag{3.165}$$

The integration over the multidimensional unit Gaussian can be written as an iterated integral over one-dimensional Gaussian distributions, and each of the one-dimensional integrals can be approximated with Gauss-Hermite quadrature:

$$\begin{aligned}
&\int \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\, \boldsymbol{\xi})\; \mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I})\, \mathrm{d}\boldsymbol{\xi} \\
&\quad = \int \cdots \int \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\, \boldsymbol{\xi})\; \mathrm{N}(\xi_1 \,|\, 0, 1)\, \mathrm{d}\xi_1 \times \cdots \times \mathrm{N}(\xi_n \,|\, 0, 1)\, \mathrm{d}\xi_n \\
&\quad \approx \sum_{i_1, \ldots, i_n} W^{(i_1)} \times \cdots \times W^{(i_n)} \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\, \boldsymbol{\xi}^{(i_1, \ldots, i_n)}).
\end{aligned} \tag{3.166}$$

The weights $W^{(i_k)}, k = 1, \ldots, n$ are simply the corresponding one-dimensional Gauss-Hermite weights and $\boldsymbol{\xi}^{(i_1,\ldots,i_n)}$ is a n-dimensional vector with one-dimensional unit sigma point $\xi^{(i_k)}$ at element k. The algorithm can be now written as follows:

Algorithm 3.24 (Gauss-Hermite cubature)**.** *The pth order Gauss-Hermite approximation to the multidimensional integral*

$$\int \mathbf{g}(\mathbf{x}) \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \, \mathrm{d}\mathbf{x} \tag{3.167}$$

can be computed as follows:

1. *Compute the one-dimensional weights $W^{(i)}, i = 1, \ldots, p$ and unit sigma points $\xi^{(i)}$ as in the one-dimensional Gauss-Hermite quadrature Algorithm 3.23.*
2. *Form multidimensional weights as the products of one-dimensional weights:*
$$\begin{aligned} W^{(i_1,\ldots,i_n)} &= W^{(i_1)} \times \cdots \times W^{(i_n)} \\ &= \frac{p!}{p^2 \, [H_{p-1}(\xi^{(i_1)})]^2} \times \cdots \times \frac{p!}{p^2 \, [H_{p-1}(\xi^{(i_n)})]^2}, \end{aligned} \tag{3.168}$$
where each i_k takes values $1, \ldots, p$.
3. *Form multidimensional unit sigma points as Cartesian product of the one-dimensional unit sigma points:*
$$\boldsymbol{\xi}^{(i_1,\ldots,i_n)} = \begin{pmatrix} \xi^{(i_1)} \\ \vdots \\ \xi^{(i_n)} \end{pmatrix}. \tag{3.169}$$
4. *Approximate the integral as*
$$\int \mathbf{g}(\mathbf{x}) \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \, \mathrm{d}\mathbf{x} \approx \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)} \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}} \, \boldsymbol{\xi}^{(i_1,\ldots,i_n)}), \tag{3.170}$$
where $\sqrt{\mathbf{P}}$ is a matrix square root defined by $\mathbf{P} = \sqrt{\mathbf{P}} \sqrt{\mathbf{P}}^T$.

The pth order multidimensional Gauss-Hermite integration is exact for monomials of the form $x^{d_1} \, x^{d_2} \cdots x^{d_n}$, and their arbitrary linear combinations, where each of the orders $d_i \leq 2p - 1$. The number of sigma points required for n-dimensional integral with pth order rule is p^n, which quickly becomes unfeasible when the number of dimensions grows.

3.3.4 Gauss-Hermite Kalman Filter (GHKF)

The additive form multidimensional Gauss-Hermite cubature based filter can be derived by replacing the Gaussian integrals in the Gaussian filter Algorithm 3.21 with the Gauss-Hermite approximations in Algorithm 3.24:

Algorithm 3.25 (Gauss-Hermite Kalman filter)**.** *The additive form Gauss-Hermite Kalman filter (GHKF) algorithm is the following:*

1. Prediction step:

   (a) *Form the sigma points as:*

$$\mathcal{X}_{k-1}^{(i_1,\ldots,i_n)} = \mathbf{m}_{k-1} + \sqrt{\mathbf{P}_{k-1}}\,\boldsymbol{\xi}^{(i_1,\ldots,i_n)} \qquad i_1,\ldots,i_n = 1,\ldots,p, \tag{3.171}$$

   where the unit sigma points $\boldsymbol{\xi}^{(i_1,\ldots,i_n)}$ *were defined in Equation* (3.169).

   (b) *Propagate the sigma points through the dynamic model:*

$$\hat{\mathcal{X}}_{k}^{(i_1,\ldots,i_n)} = \mathbf{f}(\mathcal{X}_{k-1}^{(i_1,\ldots,i_n)}), \quad i_1,\ldots,i_n = 1,\ldots,p. \tag{3.172}$$

   (c) *Compute the predicted mean* $\mathbf{m}_k^-$ *and the predicted covariance* $\mathbf{P}_k^-$*:*

$$\begin{aligned} \mathbf{m}_k^- &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)} \hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} \\ \mathbf{P}_k^- &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)} (\hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} - \mathbf{m}_k^-)\,(\hat{\mathcal{X}}_k^{(i_1,\ldots,i_n)} - \mathbf{m}_k^-)^T + \mathbf{Q}_{k-1}, \end{aligned} \tag{3.173}$$

   where the weights $W^{(i_1,\ldots,i_n)}$ *were defined in Equation* (3.168).

2. Update step:

   (a) *Form the sigma points:*

$$\mathcal{X}_{k}^{-(i_1,\ldots,i_n)} = \mathbf{m}_k^- + \sqrt{\mathbf{P}_k^-}\,\boldsymbol{\xi}^{(i_1,\ldots,i_n)}, \qquad i_1,\ldots,i_n = 1,\ldots,p, \tag{3.174}$$

   where the unit sigma points $\boldsymbol{\xi}^{(i_1,\ldots,i_n)}$ *were defined in Equation* (3.169).

   (b) *Propagate sigma points through the measurement model:*

$$\hat{\mathcal{Y}}_{k}^{(i_1,\ldots,i_n)} = \mathbf{h}(\mathcal{X}_{k}^{-(i_1,\ldots,i_n)}), \quad i_1,\ldots,i_n = 1,\ldots,p. \tag{3.175}$$

(c) *Compute the predicted mean $\boldsymbol{\mu}_k$, the predicted covariance of the measurement $\mathbf{S}_k$, and the cross-covariance of the state and the measurement $\mathbf{C}_k$:*

$$\begin{aligned}
\boldsymbol{\mu}_k &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)} \hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} \\
\mathbf{S}_k &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)} (\hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} - \boldsymbol{\mu}_k)\, (\hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} - \boldsymbol{\mu}_k)^T + \mathbf{R}_k \\
\mathbf{C}_k &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)} (\mathcal{X}_k^{-(i_1,\ldots,i_n)} - \mathbf{m}_k^-)\, (\hat{\mathcal{Y}}_k^{(i_1,\ldots,i_n)} - \boldsymbol{\mu}_k)^T,
\end{aligned} \tag{3.176}$$

where the weights $W^{(i_1,\ldots,i_n)}$ were defined in Equation (3.168).

(d) *Compute the filter gain $\mathbf{K}_k$, the filtered state mean $\mathbf{m}_k$ and the covariance $\mathbf{P}_k$, conditional on the measurement $\mathbf{y}_k$:*

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{C}_k\, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, [\mathbf{y}_k - \boldsymbol{\mu}_k] \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T.
\end{aligned} \tag{3.177}$$

The non-additive version can be obtained by applying the Gauss-Hermite quadrature to the non-additive Gaussian filter Algorithm 3.22 in a similar manner. However, due to the rapid growth of computational requirements in state dimension the augmented form is computationally quite heavy, because it requires roughly doubling of the dimensionality of the integration variable.

3.3.5 Spherical Cubature Integration

In this section we shall derive the third order spherical cubature rule, which was popularized by Arasaratnam and Haykin (2009). However, instead of using the derivation of Arasaratnam and Haykin (2009), we shall use the derivation presented by Wu et al. (2006), due to its simplicity. Although the derivation that we present here is far simpler than the alternative, it is completely equivalent. Furthermore, the derivation presented here can be more easily extended to more complicated spherical cubatures.

Recall from Section 3.3.3 that expectation of a non-linear function over an arbitrary Gaussian distribution $\mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P})$ can always be transformed into expectation over unit Gaussian distribution $\mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I})$. Thus, we can start by considering the multidimensional unit Gaussian integral

$$\int \mathbf{g}(\boldsymbol{\xi})\; \mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I})\, \mathrm{d}\boldsymbol{\xi}. \tag{3.178}$$

We now wish to form a $2n$-point approximation of the form

$$\int \mathbf{g}(\boldsymbol{\xi})\; \mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I})\, \mathrm{d}\boldsymbol{\xi} \approx W \sum_i \mathbf{g}(c\, \mathbf{u}^{(i)}), \tag{3.179}$$

where the points $\mathbf{u}^{(i)}$ belong to the symmetric set $[\mathbf{1}]$ with generator $(1, 0, \ldots, 0)$ (see, e.g., Wu et al., 2006; Arasaratnam and Haykin, 2009):

$$[\mathbf{1}] = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \cdots \begin{pmatrix} -1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \cdots \right\} \tag{3.180}$$

and W is a weight and c is a parameter yet to be determined.

Because the point set is symmetric, the rule is exact for all monomials of the form $x_1^{d_1} x_2^{d_2} \cdots x_n^{d_n}$, if at least one of the exponents d_i is odd. Thus we can construct a rule which is exact up to third degree by determining the coefficients W and c such that it is exact for selections $g_j(\boldsymbol{\xi}) = 1$ and $g_j(\boldsymbol{\xi}) = \xi_j^2$. Because the true values of the integrals are

$$\begin{aligned} \int \mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I}) \, \mathrm{d}\boldsymbol{\xi} &= 1 \\ \int \xi_j^2 \; \mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I}) \, \mathrm{d}\boldsymbol{\xi} &= 1, \end{aligned} \tag{3.181}$$

we get the equations

$$\begin{aligned} W \sum_i 1 &= W \, 2n = 1 \\ W \sum_i [c \, u_j^{(i)}]^2 &= W \, 2c^2 = 1, \end{aligned} \tag{3.182}$$

which have the solutions

$$\begin{aligned} W &= \frac{1}{2n} \\ c &= \sqrt{n}. \end{aligned} \tag{3.183}$$

That is, we get the following simple rule, which is exact for monomials up to third degree:

$$\int \mathbf{g}(\boldsymbol{\xi}) \; \mathrm{N}(\boldsymbol{\xi} \,|\, \mathbf{0}, \mathbf{I}) \, \mathrm{d}\boldsymbol{\xi} \approx \frac{1}{2n} \sum_i \mathbf{g}(\sqrt{n} \, \mathbf{u}^{(i)}), \tag{3.184}$$

We can now easily extend the method to arbitrary mean and covariance by using the change of variables in Equations (3.164) and (3.165) and the result is the following algorithm:

Algorithm 3.26 (Spherical cubature integration)**.** *The 3rd order spherical cubature approximation to the multidimensional integral*

$$\int \mathbf{g}(\mathbf{x}) \; \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \, \mathrm{d}\mathbf{x} \tag{3.185}$$

can be computed as follows:

1. *Compute the unit sigma points as*

$$\boldsymbol{\xi}^{(i)} = \begin{cases} \sqrt{n}\,\mathbf{e}_i &, \quad i = 1, \ldots, n \\ -\sqrt{n}\,\mathbf{e}_{i-n} &, \quad i = n+1, \ldots, 2n, \end{cases} \tag{3.186}$$

   where $\mathbf{e}_i$ denotes a unit vector to the direction of coordinate axis i.

2. *Approximate the integral as*

$$\int \mathbf{g}(\mathbf{x})\ \mathrm{N}(\mathbf{x}\,|\,\mathbf{m},\mathbf{P})\,\mathrm{d}\mathbf{x} \approx \frac{1}{2n}\sum_{i=1}^{2n} \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\,\boldsymbol{\xi}^{(i)}), \tag{3.187}$$

   where $\sqrt{\mathbf{P}}$ is a matrix square root defined by $\mathbf{P} = \sqrt{\mathbf{P}}\,\sqrt{\mathbf{P}}^T$.

It is easy to see that the approximation above is a special case of the unscented transform (see Section 3.2.5) with parameters $\alpha = 1$, $\beta = 0$, and $\kappa = 0$. With this parameter selection the mean weight is zero and the unscented transform is effectively a $2n$-point approximation as well.

The derivation presented by Arasaratnam and Haykin (2009) is a bit more complicated than the derivation of Wu et al. (2006) presented above, as it is based on converting the Gaussian integral into spherical coordinates and then considering the even order monomials. However, Wu et al. (2006) actually did not present the most useful special case given in the Algorithm 3.26, but instead, presented the method for more general generators $[\mathbf{u}]$. The method in the above Algorithm 3.26 has the useful property that its weights are always positive, which is not always true for more general methods (Wu et al., 2006).

We can generalize the above approach by using $2n + 1$ point approximation, where the origin is also included:

$$\int \mathbf{g}(\boldsymbol{\xi})\ \mathrm{N}(\boldsymbol{\xi}\,|\,\mathbf{0},\mathbf{I})\,\mathrm{d}\boldsymbol{\xi} \approx W_0\,\mathbf{g}(\mathbf{0}) + W\sum_i \mathbf{g}(c\,\mathbf{u}^{(i)}). \tag{3.188}$$

We can now solve the parameters W_0, W and c such that we get the exact result with selections $g_j(\boldsymbol{\xi}) = 1$ and $g_j(\boldsymbol{\xi}) = \xi_j^2$. The solution can be written in form

$$\begin{aligned} W_0 &= \frac{\kappa}{n+\kappa} \\ W &= \frac{1}{2(n+\kappa)} \\ c &= \sqrt{n+\kappa}, \end{aligned} \tag{3.189}$$

where κ is a free parameter. This gives an integration rule that can be written as

$$\int \mathbf{g}(\mathbf{x})\ \mathrm{N}(\mathbf{x}\,|\,\mathbf{m},\mathbf{P})\,\mathrm{d}\mathbf{x} \approx \frac{\kappa}{n+\kappa}\mathbf{g}(\mathbf{m}) + \frac{1}{2(n+\kappa)}\sum_{i=1}^{2n} \mathbf{g}(\mathbf{m} + \sqrt{\mathbf{P}}\,\boldsymbol{\xi}^{(i)}), \tag{3.190}$$

where

$$\boldsymbol{\xi}^{(i)} = \begin{cases} \sqrt{n+\kappa}\,\mathbf{e}_i & , \quad i = 1,\dots,n \\ -\sqrt{n+\kappa}\,\mathbf{e}_{i-n} & , \quad i = n+1,\dots,2n. \end{cases} \tag{3.191}$$

The rule can be seen to coincide with the original UT (Julier and Uhlmann, 1995), which corresponds to the unscented transform presented in Section 3.2.5 with $\alpha = 1$, $\beta = 0$ and where κ is left as a free parameter. With the selection $\kappa = 3 - n$, we can also match the fourth order moments of the distribution (Julier and Uhlmann, 1995), but with the price that when the dimensionality $n > 3$, we get negative weights and approximation rules that can sometimes be unstable. But nothing prevents us from using other values for the parameter.

Note that "third order" here means a different thing than in the Gauss-Hermite Kalman filter — a pth order Gauss-Hermite filter is exact for monomials up to order $2p-1$, which means that 3rd order GHKF is exact for monomials up to fifth order. The 3rd order spherical cubature rule is exact only for monomials up to third order. It is also possible to derive symmetric rules that are exact for higher than third order. However, this is no longer possible with a number of sigma points, which is linear $O(n)$ in state dimension (Wu et al., 2006; Arasaratnam and Haykin, 2009). For example, for fifth order rule, the required number of sigma points is proportional to n^2, the state dimension squared.

3.3.6 Cubature Kalman Filter (CKF)

When we apply the 3rd spherical cubature integration rule in Algorithm 3.26 to the Gaussian filter equations in Algorithm 3.21, we get the cubature Kalman filter (CKF) of Arasaratnam and Haykin (2009):

Algorithm 3.27 (Cubature Kalman filter I)**.** *The additive form cubature Kalman filter (CKF) algorithm is the following:*

1. Prediction step:

   (a) *Form the sigma points as:*

   $$\mathcal{X}^{(i)}_{k-1} = \mathbf{m}_{k-1} + \sqrt{\mathbf{P}_{k-1}}\,\boldsymbol{\xi}^{(i)} \qquad i = 1,\dots,2n, \tag{3.192}$$

   where the unit sigma points are defined as

   $$\boldsymbol{\xi}^{(i)} = \begin{cases} \sqrt{n}\,\mathbf{e}_i & , \quad i = 1,\dots,n \\ -\sqrt{n}\,\mathbf{e}_{i-n} & , \quad i = n+1,\dots,2n. \end{cases} \tag{3.193}$$

   (b) *Propagate the sigma points through the dynamic model:*

   $$\hat{\mathcal{X}}^{(i)}_k = \mathbf{f}(\mathcal{X}^{(i)}_{k-1}), \quad i = 1\dots 2n. \tag{3.194}$$

(c) *Compute the predicted mean $\mathbf{m}_k^-$ and the predicted covariance $\mathbf{P}_k^-$:*

$$\begin{aligned} \mathbf{m}_k^- &= \frac{1}{2n} \sum_{i=1}^{2n} \hat{\mathcal{X}}_k^{(i)} \\ \mathbf{P}_k^- &= \frac{1}{2n} \sum_{i=1}^{2n} (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-) \, (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)^T + \mathbf{Q}_{k-1}. \end{aligned} \tag{3.195}$$

2. Update step:

(a) *Form the sigma points:*

$$\mathcal{X}_k^{-(i)} = \mathbf{m}_k^- + \sqrt{\mathbf{P}_k^-} \, \boldsymbol{\xi}^{(i)}, \qquad i = 1, \ldots, 2n, \tag{3.196}$$

where the unit sigma points are defined as in Equation (3.193).

(b) *Propagate sigma points through the measurement model:*

$$\hat{\mathcal{Y}}_k^{(i)} = \mathbf{h}(\mathcal{X}_k^{-(i)}), \quad i = 1 \ldots 2n. \tag{3.197}$$

(c) *Compute the predicted mean $\boldsymbol{\mu}_k$, the predicted covariance of the measurement $\mathbf{S}_k$, and the cross-covariance of the state and the measurement $\mathbf{C}_k$:*

$$\begin{aligned} \boldsymbol{\mu}_k &= \frac{1}{2n} \sum_{i=1}^{2n} \hat{\mathcal{Y}}_k^{(i)} \\ \mathbf{S}_k &= \frac{1}{2n} \sum_{i=1}^{2n} (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k) \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T + \mathbf{R}_k \\ \mathbf{C}_k &= \frac{1}{2n} \sum_{i=1}^{2n} (\mathcal{X}_k^{-(i)} - \mathbf{m}_k^-) \, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T. \end{aligned} \tag{3.198}$$

(d) *Compute the filter gain $\mathbf{K}_k$ and the filtered state mean $\mathbf{m}_k$ and covariance $\mathbf{P}_k$, conditional on the measurement $\mathbf{y}_k$:*

$$\begin{aligned} \mathbf{K}_k &= \mathbf{C}_k \, \mathbf{S}_k^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k \, [\mathbf{y}_k - \boldsymbol{\mu}_k] \\ \mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k \, \mathbf{S}_k \, \mathbf{K}_k^T. \end{aligned} \tag{3.199}$$

By applying the cubature rule to the non-additive Gaussian filter in Algorithm 3.22 we get the following augmented form cubature Kalman filter (CKF):

Algorithm 3.28 (Cubature Kalman filter II)**.** *The augmented non-additive form cubature Kalman filter (CKF) algorithm is the following:*

1. Prediction step:

   (a) Form the matrix of sigma points for the augmented random variable $(\mathbf{x}_{k-1}, \mathbf{q}_{k-1})$*:*

$$\tilde{\mathcal{X}}_{k-1}^{(i)} = \tilde{\mathbf{m}}_{k-1} + \sqrt{\tilde{\mathbf{P}}_{k-1}}\, \boldsymbol{\xi}^{(i)'} \qquad i = 1, \ldots, 2n', \tag{3.200}$$

   where

$$\tilde{\mathbf{m}}_{k-1} = \begin{pmatrix} \mathbf{m}_{k-1} \\ \mathbf{0} \end{pmatrix} \qquad \tilde{\mathbf{P}}_{k-1} = \begin{pmatrix} \mathbf{P}_{k-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_{k-1} \end{pmatrix}.$$

   Here $n' = n + n_q$*, where* n *is the dimensionality of the state* $\mathbf{x}_{k-1}$ *and* n_q *is the dimensionality of the noise* $\mathbf{q}_{k-1}$*. The unit sigma points are defined as*

$$\boldsymbol{\xi}^{(i)'} = \begin{cases} \sqrt{n'}\, \mathbf{e}_i & , \quad i = 1, \ldots, n' \\ -\sqrt{n'}\, \mathbf{e}_{i-n'} & , \quad i = n'+1, \ldots, 2n'. \end{cases} \tag{3.201}$$

   (b) Propagate the sigma points through the dynamic model:

$$\hat{\mathcal{X}}_k^{(i)} = \mathbf{f}(\tilde{\mathcal{X}}_{k-1}^{(i),x}, \tilde{\mathcal{X}}_{k-1}^{(i),q}), \quad i = 1 \ldots 2n'. \tag{3.202}$$

   where $\tilde{\mathcal{X}}_{k-1}^{(i),x}$ *denotes the first* n *components in* $\tilde{\mathcal{X}}_{k-1}^{(i)}$ *and* $\tilde{\mathcal{X}}_{k-1}^{(i),q}$ *denotes the* n_q *last components.*

   (c) Compute the predicted mean $\mathbf{m}_k^-$ *and the predicted covariance* $\mathbf{P}_k^-$*:*

$$\begin{aligned} \mathbf{m}_k^- &= \frac{1}{2n} \sum_{i=1}^{2n'} \hat{\mathcal{X}}_k^{(i)} \\ \mathbf{P}_k^- &= \frac{1}{2n} \sum_{i=1}^{2n'} (\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)\,(\hat{\mathcal{X}}_k^{(i)} - \mathbf{m}_k^-)^T. \end{aligned} \tag{3.203}$$

2. Update step:

   (a) Let $n'' = n + n_r$*, where* n *is the dimensionality of the state and* n_r *is the dimensionality of the measurement noise. Form the sigma points for the augmented vector* $(\mathbf{x}_k, \mathbf{r}_k)$ *as follows:*

$$\tilde{\mathcal{X}}_k^{-(i)} = \tilde{\mathbf{m}}_k^- + \sqrt{\tilde{\mathbf{P}}_k^-}\, \boldsymbol{\xi}^{(i)''}, \qquad i = 1, \ldots, 2n'', \tag{3.204}$$

   where

$$\tilde{\mathbf{m}}_k^- = \begin{pmatrix} \mathbf{m}_k^- \\ \mathbf{0} \end{pmatrix} \qquad \tilde{\mathbf{P}}_k^- = \begin{pmatrix} \mathbf{P}_k^- & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_k \end{pmatrix}.$$

   The unit sigma points $\boldsymbol{\xi}^{(i)''}$ *are defined as in Equation* (3.201)*, but with* n' *replaced by* n''*.*

(b) *Propagate sigma points through the measurement model:*

$$\hat{\mathcal{Y}}_k^{(i)} = \mathbf{h}(\tilde{\mathcal{X}}_k^{-(i),x}, \tilde{\mathcal{X}}_k^{-(i),r}), \quad i = 1 \ldots 2n'', \tag{3.205}$$

where $\tilde{\mathcal{X}}_k^{-(i),x}$ *denotes the first* n *components in* $\tilde{\mathcal{X}}_k^{-(i)}$ *and* $\tilde{\mathcal{X}}_k^{-(i),r}$ *denotes the* n_r *last components.*

(c) *Compute the predicted mean* $\boldsymbol{\mu}_k$*, the predicted covariance of the measurement* $\mathbf{S}_k$*, and the cross-covariance of the state and the measurement* $\mathbf{C}_k$*:*

$$\begin{aligned}
\boldsymbol{\mu}_k &= \frac{1}{2n''} \sum_{i=1}^{2n''} \hat{\mathcal{Y}}_k^{(i)} \\
\mathbf{S}_k &= \frac{1}{2n''} \sum_{i=1}^{2n''} (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)\, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T \\
\mathbf{C}_k &= \frac{1}{2n''} \sum_{i=1}^{2n''} (\mathcal{X}_k^{-(i),x} - \mathbf{m}_k^-)\, (\hat{\mathcal{Y}}_k^{(i)} - \boldsymbol{\mu}_k)^T.
\end{aligned} \tag{3.206}$$

(d) *Compute the filter gain* $\mathbf{K}_k$*, the filtered state mean* $\mathbf{m}_k$ *and the covariance* $\mathbf{P}_k$*, conditional on the measurement* $\mathbf{y}_k$*:*

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{C}_k\, \mathbf{S}_k^{-1} \\
\mathbf{m}_k &= \mathbf{m}_k^- + \mathbf{K}_k\, [\mathbf{y}_k - \boldsymbol{\mu}_k] \\
\mathbf{P}_k &= \mathbf{P}_k^- - \mathbf{K}_k\, \mathbf{S}_k\, \mathbf{K}_k^T.
\end{aligned} \tag{3.207}$$

Note that although in cubature Kalman filter (CKF) literature the "third order" characteristic of the cubature integration rule is often emphasized (cf. Arasaratnam and Haykin, 2009), it is important to remember that in the covariance computation, the rule is only exact for first order polynomials. Thus in that sense CKF is a first order method.

3.4 Particle Filtering

Although in many filtering problems Gaussian approximations work well, sometimes the filtering distributions can be, for example, multi-modal or some of the state components might be discrete, in which cases Gaussian approximations are not appropriate. In such cases sequential importance resampling based particle filters can be a better alternative. This section is concerned with particle filters, which are methods for forming Monte Carlo approximations to the solutions of the Bayesian optimal filtering equations.

3.4.1 Monte Carlo Approximations in Bayesian Inference

In Bayesian inference, including Bayesian optimal filtering, the main inference problem can often be reduced into computation of arbitrary expectations over the posterior distribution[5]:

$$\mathrm{E}[\mathbf{g}(\mathbf{x}) \,|\, \mathbf{y}_{1:T}] = \int \mathbf{g}(\mathbf{x})\, p(\mathbf{x} \,|\, \mathbf{y}_{1:T})\, \mathrm{d}\mathbf{x}, \tag{3.208}$$

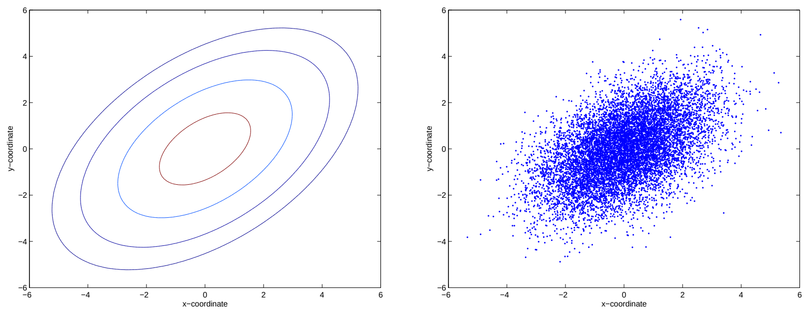
where $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$ in an arbitrary function and $p(\mathbf{x} \,|\, \mathbf{y}_{1:T})$ is the posterior probability density of $\mathbf{x}$ given the measurements $\mathbf{y}_1, \ldots, \mathbf{y}_T$. Now the problem is that such an integral can be evaluated in closed form only in a few special cases and generally, numerical methods have to be used.

Monte Carlo methods provide a numerical method for calculating integrals of the form (3.208). Monte Carlo refers to a general class of methods where closed form computation of statistical quantities is replaced by drawing samples from the distribution and estimating the quantities by sample averages.

In (perfect) Monte Carlo approximation, we draw N independent random samples from $\mathbf{x}^{(i)} \sim p(\mathbf{x} \,|\, \mathbf{y}_{1:T})$ and estimate the expectation as

$$\mathrm{E}[\mathbf{g}(\mathbf{x}) \,|\, \mathbf{y}_{1:T}] \approx \frac{1}{N} \sum_{i=1}^{N} \mathbf{g}(\mathbf{x}^{(i)}). \tag{3.209}$$

Thus Monte Carlo methods approximate the target density by a set of samples that are distributed according to the target density. Figure 3.8 represents a two dimensional Gaussian distribution and its Monte Carlo representation.



(a) (b)

Figure 3.8: (a) Two dimensional Gaussian density. (b) Monte Carlo representation of the same Gaussian density.

The convergence of Monte Carlo approximation is guaranteed by the Central Limit Theorem (CLT) (see, e.g., Liu, 2001) and the error term is $O(N^{-1/2})$, regardless of dimensionality of $\mathbf{x}$. This invariance with respect to dimensionality is

[5] In this section we formally treat $\mathbf{x}$ as a continuous random variable with a density, but the analogous results apply to discrete random variables.

unique to Monte Carlo methods and makes them superior to practically all other numerical methods when the dimensionality of $\mathbf{x}$ is considerable. At least in theory, not necessarily in practice.

3.4.2 Importance Sampling

Often in practical Bayesian models, it is not possible to obtain samples directly from $p(\mathbf{x} \,|\, \mathbf{y}_{1:T})$ due to its complicated formal appearance. In *importance sampling* (IS) (see, e.g., Liu, 2001) we use an approximate distribution called the importance distribution $\pi(\mathbf{x} \,|\, \mathbf{y}_{1:T})$, from which we can easily draw samples. Importance sampling is based on the following decomposition of the expectation over the posterior probability density $p(\mathbf{x} \,|\, \mathbf{y}_{1:T})$:

$$\int \mathbf{g}(\mathbf{x})\, p(\mathbf{x} \,|\, \mathbf{y}_{1:T})\, \mathrm{d}\mathbf{x} = \int \left[\mathbf{g}(\mathbf{x})\, \frac{p(\mathbf{x} \,|\, \mathbf{y}_{1:T})}{\pi(\mathbf{x} \,|\, \mathbf{y}_{1:T})} \right] \pi(\mathbf{x} \,|\, \mathbf{y}_{1:T})\, \mathrm{d}\mathbf{x}, \tag{3.210}$$

where the importance density $\pi(\mathbf{x} \,|\, \mathbf{y}_{1:T})$ is required to be non-zero whenever $p(\mathbf{x} \,|\, \mathbf{y}_{1:T})$ is non-zero, that is, the *support* of $\pi(\mathbf{x} \,|\, \mathbf{y}_{1:T})$ needs to be greater or equal to the support of $p(\mathbf{x} \,|\, \mathbf{y}_{1:T})$. As the above expression is just the expectation of the term in the brackets over the distribution $\pi(\mathbf{x} \,|\, \mathbf{y}_{1:T})$, we can form Monte Carlo approximation to it by drawing N samples from the importance density:

$$\mathbf{x}^{(i)} \sim \pi(\mathbf{x} \,|\, \mathbf{y}_{1:T}), \qquad i = 1, \ldots, N, \tag{3.211}$$

and by forming the approximation as

$$\begin{aligned} \mathrm{E}[\mathbf{g}(\mathbf{x}) \,|\, \mathbf{y}_{1:T}] &\approx \frac{1}{N} \sum_{i=1}^{N} \frac{p(\mathbf{x}^{(i)} \,|\, \mathbf{y}_{1:T})}{\pi(\mathbf{x}^{(i)} \,|\, \mathbf{y}_{1:T})}\, \mathbf{g}(\mathbf{x}^{(i)}) \\ &= \sum_{i=1}^{N} \tilde{w}^{(i)}\, \mathbf{g}(\mathbf{x}^{(i)}) \end{aligned} \tag{3.212}$$

where the weights have been defined as

$$\tilde{w}^{(i)} = \frac{1}{N}\, \frac{p(\mathbf{x}^{(i)} \,|\, \mathbf{y}_{1:T})}{\pi(\mathbf{x}^{(i)} \,|\, \mathbf{y}_{1:T})}. \tag{3.213}$$

Figure 3.9 illustrates the idea of importance sampling. We sample from the importance distribution, which is an approximation to the target distribution. Because the distribution of samples is not exact, we need to correct the approximation by associating a weight to each of the samples.

The disadvantage of this direct importance sampling is that we should be able to evaluate $p(\mathbf{x}^{(i)} \,|\, \mathbf{y}_{1:T})$ in order to use it directly. Recall that by Bayes' rule the posterior probability density can be written as

$$p(\mathbf{x}^{(i)} \,|\, \mathbf{y}_{1:T}) = \frac{p(\mathbf{y}_{1:T} \,|\, \mathbf{x}^{(i)})\, p(\mathbf{x}^{(i)})}{\int p(\mathbf{y}_{1:T} \,|\, \mathbf{x})\, p(\mathbf{x})\, \mathrm{d}\mathbf{x}}. \tag{3.214}$$

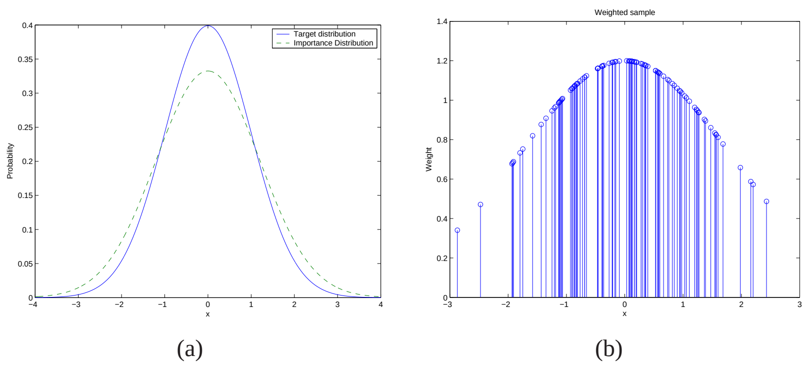


Figure 3.9: (a) Importance distribution approximates the target distribution (b) Weights are associated to each of the samples to correct the approximation.

The likelihood $p(\mathbf{y}_{1:T} \mid \mathbf{x}^{(i)})$ and prior terms $p(\mathbf{x}^{(i)})$ are usually easy to evaluate but often the integral in the denominator — the normalization constant — cannot be computed. To overcome this problem, we can form importance sampling based approximation to the expectation integral by approximating also the normalization constant by importance sampling. For this purpose we can decompose the expectation integral and form the approximation as follows:

$$\begin{aligned}
\mathrm{E}[\mathbf{g}(\mathbf{x}) \mid \mathbf{y}_{1:T}] &= \int \mathbf{g}(\mathbf{x})\, p(\mathbf{x} \mid \mathbf{y}_{1:T})\, \mathrm{d}\mathbf{x} \\
&= \frac{\int \mathbf{g}(\mathbf{x})\, p(\mathbf{y}_{1:T} \mid \mathbf{x})\, p(\mathbf{x})\, \mathrm{d}\mathbf{x}}{\int p(\mathbf{y}_{1:T} \mid \mathbf{x})\, p(\mathbf{x})\, \mathrm{d}\mathbf{x}} \\
&= \frac{\int \left[\frac{p(\mathbf{y}_{1:T} \mid \mathbf{x})\, p(\mathbf{x})}{\pi(\mathbf{x} \mid \mathbf{y}_{1:T})}\, \mathbf{g}(\mathbf{x})\right] \pi(\mathbf{x} \mid \mathbf{y}_{1:T})\, \mathrm{d}\mathbf{x}}{\int \left[\frac{p(\mathbf{y}_{1:T} \mid \mathbf{x})\, p(\mathbf{x})}{\pi(\mathbf{x} \mid \mathbf{y}_{1:T})}\right] \pi(\mathbf{x} \mid \mathbf{y}_{1:T})\, \mathrm{d}\mathbf{x}} \\
&\approx \frac{\frac{1}{N} \sum_{i=1}^{N} \frac{p(\mathbf{y}_{1:T} \mid \mathbf{x}^{(i)})\, p(\mathbf{x}^{(i)})}{\pi(\mathbf{x}^{(i)} \mid \mathbf{y}_{1:T})}\, \mathbf{g}(\mathbf{x}^{(i)})}{\frac{1}{N} \sum_{j=1}^{N} \frac{p(\mathbf{y}_{1:T} \mid \mathbf{x}^{(j)})\, p(\mathbf{x}^{(j)})}{\pi(\mathbf{x}^{(j)} \mid \mathbf{y}_{1:T})}} \\
&= \sum_{i=1}^{N} \underbrace{\left[\frac{\frac{p(\mathbf{y}_{1:T} \mid \mathbf{x}^{(i)})\, p(\mathbf{x}^{(i)})}{\pi(\mathbf{x}^{(i)} \mid \mathbf{y}_{1:T})}}{\sum_{j=1}^{N} \frac{p(\mathbf{y}_{1:T} \mid \mathbf{x}^{(j)})\, p(\mathbf{x}^{(j)})}{\pi(\mathbf{x}^{(j)} \mid \mathbf{y}_{1:T})}}\right]}_{w^{(i)}} \mathbf{g}(\mathbf{x}^{(i)}).
\end{aligned} \tag{3.215}$$

Thus we get the following algorithm:

Algorithm 3.29 (Importance sampling)**.** *Given a measurement model $p(\mathbf{y}_{1:T} \mid \mathbf{x})$ and a prior $p(\mathbf{x})$ we can form an importance sampling approximation to the posterior as follows:*

1. *Draw N samples from the importance distribution:*

$$\mathbf{x}^{(i)} \sim \pi(\mathbf{x} \,|\, \mathbf{y}_{1:T}), \qquad i = 1, \ldots, N. \tag{3.216}$$

2. *Compute the unnormalized weights by*

$$w^{*(i)} = \frac{p(\mathbf{y}_{1:T} \,|\, \mathbf{x}^{(i)})\, p(\mathbf{x}^{(i)})}{\pi(\mathbf{x}^{(i)} \,|\, \mathbf{y}_{1:T})}, \tag{3.217}$$

and the normalized weights by

$$w^{(i)} = \frac{w^{*(i)}}{\sum_{j=1}^{N} w^{*(j)}}. \tag{3.218}$$

3. *The approximation to the posterior expectation of $\mathbf{g}(\mathbf{x})$ is then given as*

$$\mathrm{E}[\mathbf{g}(\mathbf{x}) \,|\, \mathbf{y}_{1:T}] \approx \sum_{i=1}^{N} w^{(i)}\, \mathbf{g}(\mathbf{x}^{(i)}). \tag{3.219}$$

The approximation to the posterior probability density formed by the above algorithm can then be formally written as

$$p(\mathbf{x} \,|\, \mathbf{y}_{1:T}) \approx \sum_{i=1}^{N} w^{(i)}\, \delta(\mathbf{x} - \mathbf{x}^{(i)}), \tag{3.220}$$

where $\delta(\cdot)$ is the Dirac delta function.

3.4.3 Sequential Importance Sampling

Sequential importance sampling (SIS) (see, e.g., Doucet et al., 2001) is a sequential version of importance sampling. The SIS algorithm can be used for generating importance sampling approximations to filtering distributions of generic state space models of the form

$$\begin{aligned} \mathbf{x}_k &\sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \\ \mathbf{y}_k &\sim p(\mathbf{y}_k \mid \mathbf{x}_k), \end{aligned} \tag{3.221}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state at time step k and $\mathbf{y}_k \in \mathbb{R}^m$ is the measurement. The state and measurements may contain both discrete and continuous components.

The SIS algorithm uses a weighted set of *particles* $\{(w_k^{(i)}, \mathbf{x}_k^{(i)}) \,:\, i = 1, \ldots, N\}$, that is, samples from an importance distribution and their weights, for representing the filtering distribution $p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k})$ such that at every time step k the approximation to the expectation of an arbitrary function $\mathbf{g}(\mathbf{x})$ can be calculated as the weighted sample average

$$\mathrm{E}[\mathbf{g}(\mathbf{x}_k) \,|\, \mathbf{y}_{1:k}] \approx \sum_{i=1}^{N} w_k^{(i)} \mathbf{g}(\mathbf{x}_k^{(i)}). \tag{3.222}$$

Equivalently, SIS can be interpreted as forming an approximation to the filtering distribution as

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}), \tag{3.223}$$

where $\delta(\cdot)$ is the Dirac delta function.

To derive the algorithm, we consider the full posterior distribution of states $\mathbf{x}_{0:k}$ given the measurements $\mathbf{y}_{1:k}$. By using the Markov properties of the model, we get the following recursion for the posterior distribution:

$$\begin{aligned} p(\mathbf{x}_{0:k} \mid \mathbf{y}_{1:k}) &\propto p(\mathbf{y}_k \mid \mathbf{x}_{0:k}, \mathbf{y}_{1:k-1})\, p(\mathbf{x}_{0:k} \mid \mathbf{y}_{1:k-1}) \\ &= p(\mathbf{y}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k-1})\, p(\mathbf{x}_{0:k-1} \mid \mathbf{y}_{1:k-1}) \\ &= p(\mathbf{y}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{x}_{k-1})\, p(\mathbf{x}_{0:k-1} \mid \mathbf{y}_{1:k-1}). \end{aligned} \tag{3.224}$$

Using a similar rationale as in the previous section, we can now construct a importance sampling method, which draws samples from a given importance distribution $\mathbf{x}_{0:k}^{(i)} \sim \pi(\mathbf{x}_{0:k} \mid \mathbf{y}_{1:k})$ and computes the importance weights by

$$w_k^{(i)} \propto \frac{p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})\, p(\mathbf{x}_{0:k-1}^{(i)} \mid \mathbf{y}_{1:k-1})}{\pi(\mathbf{x}_{0:k}^{(i)} \mid \mathbf{y}_{1:k})}. \tag{3.225}$$

If we form the importance distribution for the states $\mathbf{x}_k$ recursively as follows:

$$\pi(\mathbf{x}_{0:k} \mid \mathbf{y}_{1:k}) = \pi(\mathbf{x}_k \mid \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k})\, \pi(\mathbf{x}_{0:k-1} \mid \mathbf{y}_{1:k-1}), \tag{3.226}$$

then the expression for the weights can be written as

$$w_k^{(i)} \propto \frac{p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})} \frac{p(\mathbf{x}_{0:k-1}^{(i)} \mid \mathbf{y}_{1:k-1})}{\pi(\mathbf{x}_{0:k-1}^{(i)} \mid \mathbf{y}_{1:k-1})}. \tag{3.227}$$

Let's now assume that we have already drawn the samples $\mathbf{x}_{0:k-1}^{(i)}$ from the importance distribution $\pi(\mathbf{x}_{0:k-1} \mid \mathbf{y}_{1:k-1})$ and computed the corresponding importance weights $w_{k-1}^{(i)}$. We can now draw samples $\mathbf{x}_{0:k}^{(i)}$ from the importance distribution $\pi(\mathbf{x}_{0:k} \mid \mathbf{y}_{1:k})$ by drawing the new state samples for the step k as $\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k \mid \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})$. The importance weights from the previous step are proportional to the last term in Equation (3.227):

$$w_{k-1}^{(i)} \propto \frac{p(\mathbf{x}_{0:k-1}^{(i)} \mid \mathbf{y}_{1:k-1})}{\pi(\mathbf{x}_{0:k-1}^{(i)} \mid \mathbf{y}_{1:k-1})}, \tag{3.228}$$

and thus the weights satisfy the recursion

$$w_k^{(i)} \propto \frac{p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})}\, w_{k-1}^{(i)}. \tag{3.229}$$

The generic sequential importance sampling algorithm can now be described as follows:

Algorithm 3.30 (Sequential importance sampling)**.** *Steps of SIS are the following:*

- *Draw N samples $\mathbf{x}_0^{(i)}$ from the prior*

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \qquad i = 1, \ldots, N, \tag{3.230}$$

  and set $w_0^{(i)} = 1/N$, for all $i = 1, \ldots, N$.

- *For each $k = 1, \ldots, T$ do the following:*

  1. *Draw samples $\mathbf{x}_k^{(i)}$ from the importance distributions*

$$\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k \,|\, \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k}), \qquad i = 1, \ldots, N. \tag{3.231}$$

  2. *Calculate new weights according to*

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k \,|\, \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \,|\, \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} \,|\, \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})} \tag{3.232}$$

     and normalize them to sum to unity.

Note that it is convenient to select the importance distribution to be Markovian in the sense that:

$$\pi(\mathbf{x}_k \,|\, \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k}) = \pi(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \mathbf{y}_{1:k}). \tag{3.233}$$

With this form of importance distribution we do not need to store the whole histories $\mathbf{x}_{0:k}^{(i)}$ in the SIS algorithm, only the current states $\mathbf{x}_k^{(i)}$. This form is also convenient in SIR, because we do not need to worry about the state histories during the resampling step as in the SIR particle smoother (see Section 4.5.1). Thus in the following section we assume that the importance distribution has indeed been selected to have the above Markovian form.

3.4.4 Sequential Importance Resampling

One problem in the SIS algorithm described in the previous section is that we easily encounter the situation that almost all the particles have zero or nearly zero weights. This is called the *degeneracy* problem in particle filtering literature and it prevented practical applications of particle filters for many years.

The degeneracy problem can be solved by using a *resampling* procedure. It refers to a procedure where we draw N new samples from the discrete distribution defined by the weights and replace the old set of N samples with this new set. This procedure can be be written as the following algorithm:

Algorithm 3.31 (Resampling)**.** *Resampling procedure can be described as follows:*

1. *Interpret each weight $w_k^{(i)}$ as the probability of obtaining the sample index i in the set $\{\mathbf{x}_k^{(i)} \mid i = 1, \ldots, N\}$.*
2. *Draw N samples from that discrete distribution and replace the old sample set with this new one.*
3. *Set all weights to the constant value $w_k^{(i)} = 1/N$.*

The idea of the resampling procedure is to remove particles with very small weights and duplicate particles with large weights. Although the theoretical distribution represented by the weighted set of samples does not change, resampling introduces additional variance to estimates. This variance introduced by the resampling procedure can be reduced by proper choice of the resampling method. The *stratified resampling* algorithm (Kitagawa, 1996) is optimal in terms of variance.

Sequential importance resampling (SIR)[6] (Gordon et al., 1993; Kitagawa, 1996; Doucet et al., 2001; Ristic et al., 2004), is a generalization of the *particle filtering* framework, in which the resampling step is included as part of the sequential importance sampling algorithm.

Usually the resampling is not performed at every time step, but only when it is actually needed. One way of implementing this is to do resampling on every nth step, where n is some predefined constant. This method has the advantage that it is unbiased. Another way, which is used here, is *adaptive resampling*. In this method, the "effective" number of particles, which is estimated from the variance of the particle weights (Liu and Chen, 1995), is used for monitoring the need for resampling. The estimate for the effective number of particles can be computed as:

$$n_{\text{eff}} \approx \frac{1}{\sum_{i=1}^{N} \left(w_k^{(i)} \right)^2}, \tag{3.234}$$

where $w_k^{(i)}$ is the normalized weight of particle i at the time step k (Liu and Chen, 1995). Resampling is performed when the effective number of particles is significantly less than the total number of particles, for example, $n_{\text{eff}} < N/10$, where N is the total number of particles.

Algorithm 3.32 (Sequential importance resampling)**.** *The sequential importance resampling (SIR) algorithm, which is also called the particle filter (PF) is the following:*

- *Draw N samples $\mathbf{x}_0^{(i)}$ from the prior*

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \qquad i = 1, \ldots, N, \tag{3.235}$$

  and set $w_0^{(i)} = 1/N$, for all $i = 1, \ldots, N$.

[6] *Sequential importance resampling (SIR)* is also often referred to as *sampling importance resampling (SIR) or* sequential importance sampling resampling (SISR).

- *For each $k = 1, \ldots, T$ do the following:*

  1. *Draw samples $\mathbf{x}_k^{(i)}$ from the importance distributions*

  $$\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_{1:k}), \qquad i = 1, \ldots, N. \tag{3.236}$$

  2. *Calculate new weights according to*

  $$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_{1:k})} \tag{3.237}$$

  and normalize them to sum to unity.

  3. *If the effective number of particles* (3.234) *is too low, perform resampling.*

The performance of the SIR algorithm depends on the quality of the importance distribution $\pi(\cdot)$, which is an approximation to the posterior distribution of states given the values at the previous step. The importance distribution should be in such functional form that we can easily draw samples from it and that it is possible to evaluate the probability densities of the sample points. *The optimal importance distribution* in terms of variance (see, e.g., Doucet et al., 2001; Ristic et al., 2004) is

$$\pi(\mathbf{x}_k \mid \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k}) = p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_k). \tag{3.238}$$

If the optimal importance distribution cannot be directly used, good importance distributions can be obtained by *local linearization* where a mixture of extended Kalman filters (EKF), unscented Kalman filters (UKF) or other types of non-linear Kalman filters are used for forming the importance distribution (Doucet et al., 2000; Van der Merwe et al., 2001). Van der Merwe et al. (2001) also suggest a Metropolis-Hastings step after (or in place of) the resampling step to smooth the resulting distribution, but from their results, it seems that this extra computation step has no significant performance effect. A particle filter with UKF importance distribution is also referred to as *unscented particle filter* (UPF). Similarly, we could call a particle filter with Gauss-Hermite Kalman filter importance distribution *Gauss-Hermite particle filter* (GHPF) and one with cubature Kalman filter importance distribution *cubature particle filter* (CPF).

By tuning the resampling algorithm to specific estimation problems and possibly changing the order of weight computation and sampling, accuracy and computational efficiency of the algorithm can be improved (Fearnhead and Clifford, 2003). An important issue is that sampling is more efficient without replacement, such that duplicate samples are not stored. There is also evidence that in some situations it is more efficient to use a simple deterministic algorithm for preserving the N most likely particles. In the article (Punskaya et al., 2002) it is shown that in digital demodulation, where the sampled space is discrete and the optimization criterion is the minimum error, the deterministic algorithm performs better.

The bootstrap filter (Gordon et al., 1993) is a variation of SIR where the dynamic model $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ is used as the importance distribution. This makes the implementation of the algorithm very easy, but due to the inefficiency of the importance distribution it may require a very large number of Monte Carlo samples for accurate estimation results. In the bootstrap filter the resampling is normally done at each time step.

Algorithm 3.33 (Bootstrap filter)**.** *The bootstrap filter algorithm is as follows:*

1. *Draw new point $\mathbf{x}_k^{(i)}$ for each point in the sample set $\{\mathbf{x}_{k-1}^{(i)}, i = 1, \ldots, N\}$ from the dynamic model:*

$$\mathbf{x}_k^{(i)} \sim p(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}), \qquad i = 1, \ldots, N. \tag{3.239}$$

2. *Calculate the weights*

$$w_k^{(i)} \propto p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)}), \qquad i = 1, \ldots, N, \tag{3.240}$$

   and normalize them to sum to unity.

3. *Do resampling.*

Another variation of sequential importance resampling is the auxiliary SIR (ASIR) filter (Pitt and Shephard, 1999). The idea of the ASIR is to mimic the availability of the optimal importance distribution by performing the resampling at step $k-1$ using the available measurement at time k.

One problem encountered in particle filtering, even when using a resampling procedure, is called *sample impoverishment* (see, e.g., Ristic et al., 2004). It refers to the effect that when the noise in the dynamic model is very small, many of the particles in the particle set will turn out to have exactly the same value. That is, the resampling step simply multiplies a few (or one) particles and thus we end up having a set of identical copies of certain high weighted particles. This problem can be diminished by using, for example, the resample-move algorithm, regularization or MCMC steps (Ristic et al., 2004).

Because low noise in the dynamic model causes sample impoverishment, it also implies that pure recursive estimation with particle filters is challenging. This is because in pure recursive estimation the process noise is formally zero and thus a basic SIR based particle filter is likely to perform very badly. However, pure recursive estimation, such as recursive estimation of static parameters can sometimes be done by applying a Rao-Blackwellized particle filter instead of a basic SIR particle filter.

3.4.5 Rao-Blackwellized Particle Filter

One way of improving the efficiency of SIR is to use Rao-Blackwellization. The idea of the *Rao-Blackwellized particle filter* (RBPF) (Akashi and Kumamoto, 1977;

Doucet et al., 2001; Ristic et al., 2004), which is also called *mixture Kalman filter* (MKF) (Chen and Liu, 2000) is that sometimes it is possible to evaluate some of the filtering equations analytically and the others with Monte Carlo sampling instead of computing everything with pure sampling. According to the *Rao-Blackwell theorem* (see, e.g., Berger, 1985; Casella and Robert, 1996) this leads to estimators with less variance than could be obtained with pure Monte Carlo sampling. An intuitive way of understanding this is that the marginalization replaces the finite Monte Carlo particle set representation with an infinite closed form particle set, which is always more accurate than any finite set.

Most commonly Rao-Blackwellized particle filtering refers to marginalized filtering of conditionally Gaussian Markov models of the form

$$\begin{aligned} p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \boldsymbol{\theta}_{k-1}) &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{A}_{k-1}(\boldsymbol{\theta}_{k-1})\, \mathbf{x}_{k-1}, \mathbf{Q}_{k-1}(\boldsymbol{\theta}_{k-1})) \\ p(\mathbf{y}_k \,|\, \mathbf{x}_k, \boldsymbol{\theta}_k) &= \mathrm{N}(\mathbf{y}_k \,|\, \mathbf{H}_k(\boldsymbol{\theta}_k)\, \mathbf{x}_k, \mathbf{R}_k(\boldsymbol{\theta}_k)) \\ p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{k-1}) &= \text{(any given form)}, \end{aligned} \tag{3.241}$$

where $\mathbf{x}_k$ is the state, $\mathbf{y}_k$ is the measurement, and $\boldsymbol{\theta}_k$ is an arbitrary latent variable. If also the prior of $\mathbf{x}_k$ is Gaussian, then due to the conditionally Gaussian structure of the model the state variables $\mathbf{x}_k$ can be integrated out analytically and only the latent variables $\boldsymbol{\theta}_k$ need to be sampled. The Rao-Blackwellized particle filter uses SIR for the latent variables and computes the conditionally Gaussian part in closed form.

To derive the filtering algorithm, first note that the full posterior distribution at step k can be factored as

$$p(\boldsymbol{\theta}_{0:k}, \mathbf{x}_{0:k} \,|\, \mathbf{y}_{1:k}) = p(\mathbf{x}_{0:k} \,|\, \boldsymbol{\theta}_{0:k}, \mathbf{y}_{1:k})\, p(\boldsymbol{\theta}_{0:k} \,|\, \mathbf{y}_{1:k}), \tag{3.242}$$

where the first term is Gaussian and computable with Kalman filter and RTS smoother. For the second term we get the following recursion analogously to Equation (3.224):

$$\begin{aligned} p(\boldsymbol{\theta}_{0:k} \,|\, \mathbf{y}_{1:k}) &\propto p(\mathbf{y}_k \,|\, \boldsymbol{\theta}_{0:k}, \mathbf{y}_{1:k-1})\, p(\boldsymbol{\theta}_{0:k} \,|\, \mathbf{y}_{1:k-1}) \\ &= p(\mathbf{y}_k \,|\, \boldsymbol{\theta}_{0:k}, \mathbf{y}_{1:k-1})\, p(\boldsymbol{\theta}_k \,|\, \boldsymbol{\theta}_{0:k-1}, \mathbf{y}_{1:k-1})\, p(\boldsymbol{\theta}_{0:k-1} \,|\, \mathbf{y}_{1:k-1}) \\ &= p(\mathbf{y}_k \,|\, \boldsymbol{\theta}_{0:k}, \mathbf{y}_{1:k-1})\, p(\boldsymbol{\theta}_k \,|\, \boldsymbol{\theta}_{k-1})\, p(\boldsymbol{\theta}_{0:k-1} \,|\, \mathbf{y}_{1:k-1}), \end{aligned} \tag{3.243}$$

where we have used the Markovianity of $\boldsymbol{\theta}_k$. Now the measurements are not conditionally independent given $\boldsymbol{\theta}_k$ and thus the first term differs from the corresponding term in Equation (3.224). The first term can be computed by running Kalman filter with fixed $\boldsymbol{\theta}_{0:k}$ over the measurement sequence. The second term is just the dynamic model and the third term is the posterior from the previous step.

If we form the importance distribution recursively as follows:

$$\pi(\boldsymbol{\theta}_{0:k} \,|\, \mathbf{y}_{1:k}) = \pi(\boldsymbol{\theta}_k \,|\, \boldsymbol{\theta}_{0:k-1}, \mathbf{y}_{1:k})\, \pi(\boldsymbol{\theta}_{0:k-1} \,|\, \mathbf{y}_{1:k-1}), \tag{3.244}$$

then by following the same derivation as in Section 3.4.3, we get the following recursion for the weights

$$w_k^{(i)} \propto \frac{p(\mathbf{y}_k \mid \boldsymbol{\theta}_{0:k-1}^{(i)}, \mathbf{y}_{1:k-1})\, p(\boldsymbol{\theta}_k^{(i)} \mid \boldsymbol{\theta}_{k-1}^{(i)})}{\pi(\boldsymbol{\theta}_k^{(i)} \mid \boldsymbol{\theta}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})}\, w_{k-1}^{(i)}, \tag{3.245}$$

which corresponds to Equation (3.229). Thus via the above recursion we can form an importance sampling based approximation to the marginal distribution $p(\boldsymbol{\theta}_{0:k} \mid \mathbf{y}_{1:k})$. But because given $\boldsymbol{\theta}_{0:k}$, the distribution $p(\mathbf{x}_{0:k} \mid \boldsymbol{\theta}_{0:k}, \mathbf{y}_{1:k})$ is Gaussian, we can form the full posterior distribution by using Equation (3.242). Computing the distribution jointly for the full history $\mathbf{x}_{0:k}$ would require running both the Kalman filter and the RTS smoother over the sequences $\boldsymbol{\theta}_{0:k}$ and $\mathbf{y}_{1:k}$, but if we are only interested in the posterior of the last time step $\mathbf{x}_k$, we only need to run the Kalman filter. The resulting algorithm is the following:

Algorithm 3.34 (Conditionally Gaussian Rao-Blackwellized particle filter)**.** *Given a sequence of importance distributions $\pi(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})$ and a set of weighted samples $\{w_{k-1}^{(i)}, \boldsymbol{\theta}_{k-1}^{(i)}, \mathbf{m}_{k-1}^{(i)}, \mathbf{P}_{k-1}^{(i)} \;:\; i = 1, \ldots, N\}$, the Rao-Blackwellized particle filter (RBPF) processes the measurement $\mathbf{y}_k$ as follows (Doucet et al., 2001):*

1. *Perform Kalman filter predictions for each of the Kalman filter means and covariances in the particles $i = 1, \ldots, N$ conditional on the previously drawn latent variable values $\boldsymbol{\theta}_{k-1}^{(i)}$*

$$\begin{aligned} \mathbf{m}_k^{-(i)} &= \mathbf{A}_{k-1}(\boldsymbol{\theta}_{k-1}^{(i)})\, \mathbf{m}_{k-1}^{(i)} \\ \mathbf{P}_k^{-(i)} &= \mathbf{A}_{k-1}(\boldsymbol{\theta}_{k-1}^{(i)})\, \mathbf{P}_{k-1}^{(i)}\, \mathbf{A}_{k-1}^T(\boldsymbol{\theta}_{k-1}^{(i)}) + \mathbf{Q}_{k-1}(\boldsymbol{\theta}_{k-1}^{(i)}). \end{aligned} \tag{3.246}$$

2. *Draw new latent variables $\boldsymbol{\theta}_k^{(i)}$ for each particle in $i = 1, \ldots, N$ from the corresponding importance distributions*

$$\boldsymbol{\theta}_k^{(i)} \sim \pi(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{0:k-1}^{(i)}, \mathbf{y}_{1:k}). \tag{3.247}$$

3. *Calculate new weights as follows:*

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k \mid \boldsymbol{\theta}_{0:k}^{(i)}, \mathbf{y}_{1:k-1})\, p(\boldsymbol{\theta}_k^{(i)} \mid \boldsymbol{\theta}_{k-1}^{(i)})}{\pi(\boldsymbol{\theta}_k^{(i)} \mid \boldsymbol{\theta}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})}, \tag{3.248}$$

where the likelihood term is the marginal measurement likelihood of the Kalman filter

$$\begin{aligned} &p(\mathbf{y}_k \mid \boldsymbol{\theta}_{0:k}^{(i)}, \mathbf{y}_{1:k-1}) \\ &\quad = \mathrm{N}\left(\mathbf{y}_k \,\middle|\, \mathbf{H}_k(\boldsymbol{\theta}_k^{(i)})\, \mathbf{m}_k^{-(i)}, \mathbf{H}_k(\boldsymbol{\theta}_k^{(i)})\, \mathbf{P}_k^{-(i)}\, \mathbf{H}_k^T(\boldsymbol{\theta}_k^{(i)}) + \mathbf{R}_k(\boldsymbol{\theta}_k^{(i)})\right). \end{aligned} \tag{3.249}$$

such that the model parameters in the Kalman filter are conditioned on the drawn latent variable value $\boldsymbol{\theta}_k^{(i)}$. *Then normalize the weights to sum to unity.*

4. *Perform Kalman filter updates for each of the particles conditional on the drawn latent variables* $\boldsymbol{\theta}_k^{(i)}$

$$\begin{aligned}
\mathbf{v}_k^{(i)} &= \mathbf{y}_k - \mathbf{H}_k(\boldsymbol{\theta}_k^{(i)})\,\mathbf{m}_k^{-} \\
\mathbf{S}_k^{(i)} &= \mathbf{H}_k(\boldsymbol{\theta}_k^{(i)})\,\mathbf{P}_k^{-(i)}\,\mathbf{H}_k^T(\boldsymbol{\theta}_k^{(i)}) + \mathbf{R}_k(\boldsymbol{\theta}_k^{(i)}) \\
\mathbf{K}_k^{(i)} &= \mathbf{P}_k^{-(i)}\,\mathbf{H}_k^T(\boldsymbol{\theta}_k^{(i)})\,\mathbf{S}_k^{-1} \\
\mathbf{m}_k^{(i)} &= \mathbf{m}_k^{-(i)} + \mathbf{K}_k^{(i)}\,\mathbf{v}_k^{(i)} \\
\mathbf{P}_k^{(i)} &= \mathbf{P}_k^{-(i)} - \mathbf{K}_k^{(i)}\,\mathbf{S}_k^{(i)}\,[\mathbf{K}_k^{(i)}]^T.
\end{aligned} \tag{3.250}$$

5. *If the effective number of particles* (3.234) *is too low, perform* resampling.

The Rao-Blackwellized particle filter produces for each time step k a set of weighted samples $\{w_k^{(i)}, \boldsymbol{\theta}_k^{(i)}, \mathbf{m}_k^{(i)}, \mathbf{P}_k^{(i)} \,:\, i = 1, \ldots, N\}$ such that expectation of a function $\mathbf{g}(\cdot)$ can be approximated as

$$\mathrm{E}[\mathbf{g}(\mathbf{x}_k, \boldsymbol{\theta}_k) \mid \mathbf{y}_{1:k}] \approx \sum_{i=1}^{N} w_k^{(i)} \int \mathbf{g}(\mathbf{x}_k, \boldsymbol{\theta}_k^{(i)})\ \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_k^{(i)}, \mathbf{P}_k^{(i)})\,\mathrm{d}\mathbf{x}_k. \tag{3.251}$$

Equivalently, RBPF can be interpreted to form an approximation to the filtering distribution as

$$p(\mathbf{x}_k, \boldsymbol{\theta}_k \mid \mathbf{y}_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)}\,\delta(\boldsymbol{\theta}_k - \boldsymbol{\theta}_k^{(i)})\ \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_k^{(i)}, \mathbf{P}_k^{(i)}). \tag{3.252}$$

The optimal importance distribution, that is, the importance distribution that minimizes the variance of the importance weights in the RBPF case is

$$p(\boldsymbol{\theta}_k \mid \mathbf{y}_{1:k}, \boldsymbol{\theta}_{0:k}^{(i)}) \propto p(\mathbf{y}_k \mid \boldsymbol{\theta}_k, \boldsymbol{\theta}_{0:k-1}^{(i)})\,p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{0:k-1}^{(i)}, \mathbf{y}_{1:k-1}). \tag{3.253}$$

In general, normalizing this distribution or drawing samples from this distribution directly is not possible. But, if the latent variables $\boldsymbol{\theta}_k$ are discrete, we can normalize this distribution and use this optimal importance distribution directly.

The class models, where Rao-Blackwellization of some linear state components can be carried out, can be extended beyond the conditionally Gaussian models presented here. We can, for example, include additional latent-variable dependent non-linear terms into the dynamic and measurement models (Schön et al., 2005). In some cases, when the filtering model is not strictly Gaussian due to slight non-linearities in either dynamic or measurement models, it is possible to replace the exact Kalman filter update and prediction steps in RBPF with extended Kalman filter (EKF) or unscented Kalman filter (UKF) prediction and update steps, or with any other Gaussian approximation based filters. This approximate Rao-Blackwellization approach has been used, for example, by Särkkä et al. (2007b).

Another general class of models where Rao-Blackwellization can often be applied are state space models with unknown static parameters. These models are of the form (Storvik, 2002)

$$\begin{aligned} \mathbf{x}_k &\sim p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}, \boldsymbol{\theta}) \\ \mathbf{y}_k &\sim p(\mathbf{y}_k \,|\, \mathbf{x}_k, \boldsymbol{\theta}) \\ \boldsymbol{\theta} &\sim p(\boldsymbol{\theta}), \end{aligned} \tag{3.254}$$

where vector $\boldsymbol{\theta}$ contains the unknown static parameters. If the posterior distribution of parameters $\boldsymbol{\theta}$ depends only on some sufficient statistics

$$\mathbf{T}_k = \mathbf{T}_k(\mathbf{x}_{1:k}, \mathbf{y}_{1:k}), \tag{3.255}$$

and if the sufficient statistics are easy to update recursively, then sampling of the state and parameters can be efficiently performed by recursively computing the sufficient statistics conditionally on the sampled states and the measurements (Storvik, 2002). This idea can be extended to time-varying case if the dynamic model has such a form which keeps the predicted distribution within the same class of distributions.

A particularly useful special case is obtained when the dynamic model is independent of the parameters $\boldsymbol{\theta}$. In this case, if conditionally to the state $\mathbf{x}_k$ the prior $p(\boldsymbol{\theta})$ belongs to the conjugate family of the likelihood $p(\mathbf{y}_k \,|\, \mathbf{x}_k, \boldsymbol{\theta})$, the static parameters $\boldsymbol{\theta}$ can be marginalized out and only the states need to be sampled.

Chapter 4

Optimal Smoothing

In this chapter we shall first present the Bayesian theory of smoothing. Then we shall present the classical Rauch-Tung-Striebel smoother and its linearization based non-linear extensions. We shall also cover unscented transform, Gauss-Hermite, and cubature based non-linear RTS smoothers as well as some particle smoothers.

In addition to the various articles cited in the text, the following books contain useful information on non-linear smoothing:

- Linear smoothing can be found in classic books: Meditch (1969); Anderson and Moore (1979); Maybeck (1982a); Lewis (1986).

- Linear and non-linear case is treated, for example in the following classic books: Lee (1964); Sage and Melsa (1971); Gelb (1974) as well as in the more recent book of Crassidis and Junkins (2004).

4.1 Formal Equations and Exact Solutions

4.1.1 Optimal Smoothing Equations

The purpose of *optimal smoothing*[1] is to compute the marginal posterior distribution of the state $\mathbf{x}_k$ at the time step k after receiving the measurements up to a time step T, where $T > k$:

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:T}). \tag{4.1}$$

The difference between filters and smoothers is that *the optimal filter* computes its estimates using only the measurements obtained before and on the time step k, but *the optimal smoother* uses also the future measurements for computing its estimates. After obtaining the filtering posterior state distributions, the following theorem gives the equations for computing the marginal posterior distributions for each time step conditionally on all the measurements up to the time step T:

[1]This definition actually applies to fixed-interval type of smoothing.

Theorem 4.1 (Bayesian optimal smoothing equations)**.** *The backward recursive equations for computing the* smoothed distributions $p(\mathbf{x}_k \,|\, \mathbf{y}_{1:T})$ *for any* $k < T$ *are given by the following* Bayesian (fixed interval) smoothing equations

$$\begin{aligned} p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k}) &= \int p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k})\, \mathrm{d}\mathbf{x}_k \\ p(\mathbf{x}_k \,|\, \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) \int \left[\frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k)\, p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T})}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})} \right] \mathrm{d}\mathbf{x}_{k+1}, \end{aligned} \tag{4.2}$$

where $p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k})$ *is the filtering distribution of the time step* k*. Note that the term* $p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})$ *is simply the predicted distribution of time step* $k+1$*. The integrations are replaced by summations if some of the state components are discrete.*

Proof. Due to the Markov properties the state $\mathbf{x}_k$ is independent of $\mathbf{y}_{k+1:T}$ given $\mathbf{x}_{k+1}$, which gives $p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T}) = p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:k})$. By using *Bayes' rule* the distribution of $\mathbf{x}_k$ given $\mathbf{x}_{k+1}$ and $\mathbf{y}_{1:T}$ can be expressed as

$$\begin{aligned} p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:k}) \\ &= \frac{p(\mathbf{x}_k, \mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})} \\ &= \frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k, \mathbf{y}_{1:k})\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k})}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})} \\ &= \frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k})}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})}. \end{aligned} \tag{4.3}$$

The joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ given $\mathbf{y}_{1:T}$ can be now computed as

$$\begin{aligned} p(\mathbf{x}_k, \mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T})\, p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) \\ &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:k})\, p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) \\ &= \frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k})\, p(\mathbf{x}_{k+1} \,|\mathbf{y}_{1:T})}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})}, \end{aligned} \tag{4.4}$$

where $p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T})$ is the smoothed distribution of the time step $k+1$. The marginal distribution of $\mathbf{x}_k$ given $\mathbf{y}_{1:T}$ is given by integration (or summation) over $\mathbf{x}_{k+1}$ in Equation (4.4), which gives the desired result. □

4.1.2 Rauch-Tung-Striebel Smoother

The *Rauch-Tung-Striebel (RTS) smoother*[2] (see, e.g., Rauch et al., 1965; Gelb, 1974; Bar-Shalom et al., 2001) can be used for computing the closed form smoothing solution

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:T}) = \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k^s, \mathbf{P}_k^s), \tag{4.5}$$

[2]Also sometimes called Kalman smoother.

to the linear filtering model (3.17). The difference to the solution computed by the *Kalman filter* is that the smoothed solution is conditional on the whole measurement data $\mathbf{y}_{1:T}$, while the filtering solution is conditional only on the measurements obtained before and on the time step k, that is, on the measurements $\mathbf{y}_{1:k}$.

Theorem 4.2 (RTS smoother)**.** *The backward recursion equations for the discrete-time fixed interval Rauch-Tung-Striebel smoother (Kalman smoother) are given as*

$$\begin{aligned} \mathbf{m}_{k+1}^- &= \mathbf{A}_k\,\mathbf{m}_k \\ \mathbf{P}_{k+1}^- &= \mathbf{A}_k\,\mathbf{P}_k\,\mathbf{A}_k^T + \mathbf{Q}_k \\ \mathbf{G}_k &= \mathbf{P}_k\,\mathbf{A}_k^T\,[\mathbf{P}_{k+1}^-]^{-1} \\ \mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\,[\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-] \\ \mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\,[\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-]\,\mathbf{G}_k^T, \end{aligned} \tag{4.6}$$

where $\mathbf{m}_k$ and $\mathbf{P}_k$ are the mean and covariance computed by the Kalman filter. The recursion is started from the last time step T, with $\mathbf{m}_T^s = \mathbf{m}_T$ and $\mathbf{P}_T^s = \mathbf{P}_T$. Note that the first two of the equations are simply the Kalman filter prediction equations.

Proof. Similarly to the Kalman filter case, by Lemma A.1, the joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ given $\mathbf{y}_{1:k}$ is

$$\begin{aligned} p(\mathbf{x}_k, \mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k}) &= p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k)\, p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) \\ &= \mathrm{N}(\mathbf{x}_{k+1} \,|\, \mathbf{A}_k\,\mathbf{x}_k, \mathbf{Q}_k)\ \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k) \\ &= \mathrm{N}\left(\begin{bmatrix} \mathbf{x}_k \\ \mathbf{x}_{k+1} \end{bmatrix} \,\middle|\, \mathbf{m}_1, \mathbf{P}_1 \right), \end{aligned} \tag{4.7}$$

where

$$\mathbf{m}_1 = \begin{pmatrix} \mathbf{m}_k \\ \mathbf{A}_k\,\mathbf{m}_k \end{pmatrix}, \qquad \mathbf{P}_1 = \begin{pmatrix} \mathbf{P}_k & \mathbf{P}_k\,\mathbf{A}_k^T \\ \mathbf{A}_k\,\mathbf{P}_k & \mathbf{A}_k\,\mathbf{P}_k\,\mathbf{A}_k^T + \mathbf{Q}_k \end{pmatrix}. \tag{4.8}$$

Due to the Markov property of the states we have

$$p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T}) = p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:k}), \tag{4.9}$$

and thus by Lemma A.2 we get the conditional distribution

$$\begin{aligned} p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:k}) \\ &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_2, \mathbf{P}_2), \end{aligned} \tag{4.10}$$

where

$$\begin{aligned} \mathbf{G}_k &= \mathbf{P}_k\,\mathbf{A}_k^T\,(\mathbf{A}_k\,\mathbf{P}_k\,\mathbf{A}_k^T + \mathbf{Q}_k)^{-1} \\ \mathbf{m}_2 &= \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{x}_{k+1} - \mathbf{A}_k\,\mathbf{m}_k) \\ \mathbf{P}_2 &= \mathbf{P}_k - \mathbf{G}_k\,(\mathbf{A}_k\,\mathbf{P}_k\,\mathbf{A}_k^T + \mathbf{Q}_k)\,\mathbf{G}_k^T. \end{aligned} \tag{4.11}$$

The joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ given all the data is

$$\begin{aligned} p(\mathbf{x}_{k+1}, \mathbf{x}_k \,|\, \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T})\, p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) \\ &= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_2, \mathbf{P}_2)\; \mathrm{N}(\mathbf{x}_{k+1} \,|\, \mathbf{m}^s_{k+1}, \mathbf{P}^s_{k+1}) \\ &= \mathrm{N}\left(\begin{bmatrix} \mathbf{x}_{k+1} \\ \mathbf{x}_k \end{bmatrix} \,\Big|\, \mathbf{m}_3, \mathbf{P}_3 \right) \end{aligned} \tag{4.12}$$

where

$$\begin{aligned} \mathbf{m}_3 &= \begin{pmatrix} \mathbf{m}^s_{k+1} \\ \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{m}^s_{k+1} - \mathbf{A}_k\,\mathbf{m}_k) \end{pmatrix} \\ \mathbf{P}_3 &= \begin{pmatrix} \mathbf{P}^s_{k+1} & \mathbf{P}^s_{k+1}\,\mathbf{G}_k^T \\ \mathbf{G}_k\,\mathbf{P}^s_{k+1} & \mathbf{G}_k\,\mathbf{P}^s_{k+1}\,\mathbf{G}_k^T + \mathbf{P}_2 \end{pmatrix}. \end{aligned} \tag{4.13}$$

Thus by Lemma A.2, the marginal distribution of $\mathbf{x}_k$ is given as

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:T}) = \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}^s_k, \mathbf{P}^s_k), \tag{4.14}$$

where

$$\begin{aligned} \mathbf{m}^s_k &= \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{m}^s_{k+1} - \mathbf{A}_k\,\mathbf{m}_k) \\ \mathbf{P}^s_k &= \mathbf{P}_k + \mathbf{G}_k\,(\mathbf{P}^s_{k+1} - \mathbf{A}_k\,\mathbf{P}_k\,\mathbf{A}_k^T - \mathbf{Q}_k)\,\mathbf{G}_k^T. \end{aligned} \tag{4.15}$$

□

Example 4.1 (RTS smoother for Gaussian random walk)**.** *The RTS smoother for the random walk model given in Example 3.1 is given by the equations*

$$\begin{aligned} m^-_{k+1} &= m_k \\ P^-_{k+1} &= P_k + q \\ m^s_k &= m_k + \frac{P_k}{P^-_{k+1}}(m^s_{k+1} - m^-_{k+1}) \\ P^s_k &= P_k + \left(\frac{P_k}{P^-_{k+1}}\right)^2 [P^s_{k+1} - P^-_{k+1}], \end{aligned} \tag{4.16}$$

where m_k and P_k are the updated mean and covariance from the Kalman filter in Example 3.2.

4.2 Extended and Unscented Smoothing

4.2.1 Extended Rauch-Tung-Striebel Smoother

The first order (i.e., linearized) extended Rauch-Tung-Striebel smoother (ERTSS) (Cox, 1964; Sage and Melsa, 1971) can be obtained from the basic RTS smoother equations by replacing the prediction equations with first order approximations.

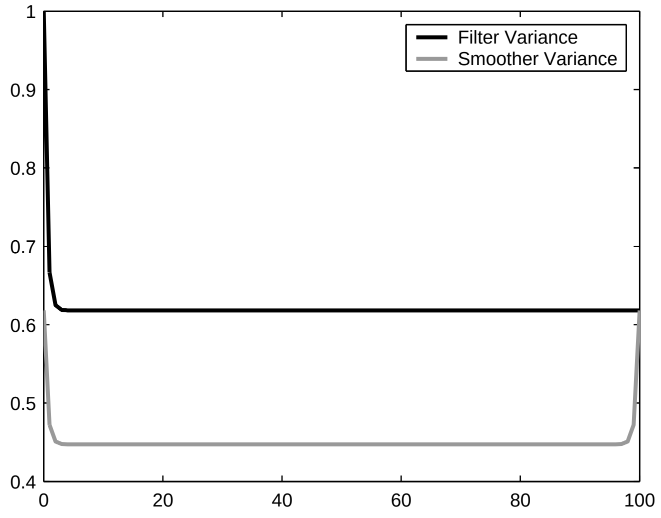


Figure 4.1: Filter and smoother variances in the Kalman smoothing example (Example 4.1).

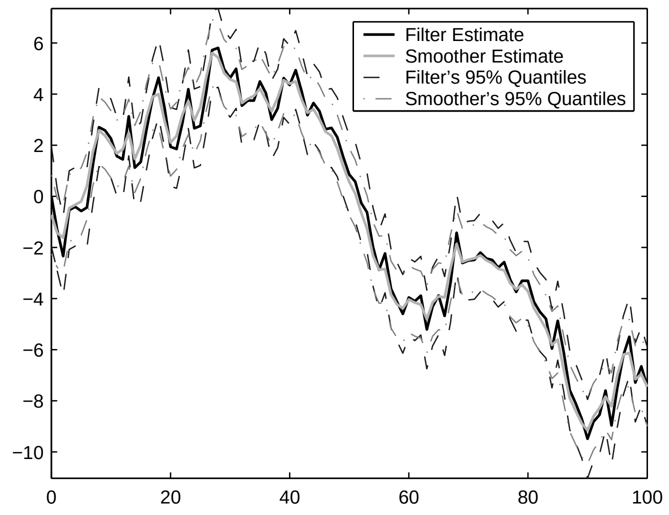


Figure 4.2: Filter and smoother estimates in the Kalman smoothing example (Example 4.1).

Higher order extended Kalman smoothers are also possible (see, e.g., Cox, 1964; Sage and Melsa, 1971), but only the first order version is presented here.

For the additive model Equation (3.55) the extended Rauch-Tung-Striebel smoother algorithm is the following:

Algorithm 4.1 (Extended RTS smoother)**.** *The equations for the extended RTS* *smoother are*

$$\begin{aligned}
\mathbf{m}_{k+1}^{-} &= \mathbf{f}(\mathbf{m}_k) \\
\mathbf{P}_{k+1}^{-} &= \mathbf{F}_{\mathbf{x}}(\mathbf{m}_k)\,\mathbf{P}_k\,\mathbf{F}_{\mathbf{x}}^T(\mathbf{m}_k) + \mathbf{Q}_k \\
\mathbf{G}_k &= \mathbf{P}_k\,\mathbf{F}_{\mathbf{x}}^T(\mathbf{m}_k)\,[\mathbf{P}_{k+1}^{-}]^{-1} \\
\mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\,[\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^{-}] \\
\mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\,[\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^{-}]\,\mathbf{G}_k^T,
\end{aligned} \tag{4.17}$$

where the matrix $\mathbf{F}_{\mathbf{x}}(\mathbf{m}_k)$ is the Jacobian matrix of $\mathbf{f}(\mathbf{x})$ evaluated at $\mathbf{m}_k$.

The above procedure is a recursion which can be used for computing the smoothing distribution of step k from the smoothing distribution of time step $k+1$. Because the smoothing distribution and filtering distribution of the last time step T are the same, we have $\mathbf{m}_T^s = \mathbf{m}_T$, $\mathbf{P}_T^s = \mathbf{P}_T$, and thus the recursion can be used for computing the smoothing distributions of all time steps by starting from the last step $k = T$ and proceeding backwards to the initial step $k = 0$.

Proof. Assume that the approximate means and covariances of the filtering distributions

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) \approx \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k),$$

for the model (3.55) have been computed by the extended Kalman filter or a similar method. Further assume that the smoothing distribution of time step $k+1$ is known and approximately Gaussian

$$p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) \approx \mathrm{N}(\mathbf{x}_{k+1} \,|\, \mathbf{m}_{k+1}^s, \mathbf{P}_{k+1}^s).$$

As in the derivation of the prediction step of EKF in Section 3.2.2, the approximate joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ given $\mathbf{y}_{1:k}$ is

$$p(\mathbf{x}_k, \mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k}) = \mathrm{N}\left(\begin{bmatrix} \mathbf{x}_k \\ \mathbf{x}_{k+1} \end{bmatrix} \,\middle|\, \mathbf{m}_1, \mathbf{P}_1 \right), \tag{4.18}$$

where

$$\begin{aligned}
\mathbf{m}_1 &= \begin{pmatrix} \mathbf{m}_k \\ \mathbf{f}(\mathbf{m}_k) \end{pmatrix} \\
\mathbf{P}_1 &= \begin{pmatrix} \mathbf{P}_k & \mathbf{P}_k\,\mathbf{F}_x^T \\ \mathbf{F}_x\,\mathbf{P}_k & \mathbf{F}_x\,\mathbf{P}_k\,\mathbf{F}_x^T + \mathbf{Q}_k \end{pmatrix}.
\end{aligned} \tag{4.19}$$

where the Jacobian matrix $\mathbf{F}_x$ of $\mathbf{f}(\mathbf{x})$ is evaluated at $\mathbf{x} = \mathbf{m}_k$. By conditioning on $\mathbf{x}_{k+1}$ as in RTS derivation in Section 4.1.2 we get

$$\begin{aligned}
p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:k}) \\
&= \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_2, \mathbf{P}_2),
\end{aligned} \tag{4.20}$$

where

$$\begin{aligned}\mathbf{G}_k &= \mathbf{P}_k\,\mathbf{F}_x^T\,(\mathbf{F}_x\,\mathbf{P}_k\,\mathbf{F}_x^T + \mathbf{Q}_k)^{-1}\\ \mathbf{m}_2 &= \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{x}_{k+1} - \mathbf{f}(\mathbf{m}_k))\\ \mathbf{P}_2 &= \mathbf{P}_k - \mathbf{G}_k\,(\mathbf{F}_x\,\mathbf{P}_k\,\mathbf{F}_x^T + \mathbf{Q}_k)\,\mathbf{G}_k^T.\end{aligned} \tag{4.21}$$

The joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ given all the data is now

$$\begin{aligned}p(\mathbf{x}_{k+1},\mathbf{x}_k\,|\,\mathbf{y}_{1:T}) &= p(\mathbf{x}_k\,|\,\mathbf{x}_{k+1},\mathbf{y}_{1:T})\,p(\mathbf{x}_{k+1}\,|\,\mathbf{y}_{1:T})\\ &= \mathrm{N}\left(\begin{bmatrix}\mathbf{x}_{k+1}\\ \mathbf{x}_k\end{bmatrix}\,\middle|\,\mathbf{m}_3,\mathbf{P}_3\right)\end{aligned} \tag{4.22}$$

where

$$\begin{aligned}\mathbf{m}_3 &= \begin{pmatrix}\mathbf{m}^s_{k+1}\\ \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{m}^s_{k+1} - \mathbf{f}(\mathbf{m}_k))\end{pmatrix}\\ \mathbf{P}_3 &= \begin{pmatrix}\mathbf{P}^s_{k+1} & \mathbf{P}^s_{k+1}\,\mathbf{G}_k^T\\ \mathbf{G}_k\,\mathbf{P}^s_{k+1} & \mathbf{G}_k\,\mathbf{P}^s_{k+1}\,\mathbf{G}_k^T + \mathbf{P}_2\end{pmatrix}.\end{aligned} \tag{4.23}$$

The marginal distribution of $\mathbf{x}_k$ is then

$$p(\mathbf{x}_k\,|\,\mathbf{y}_{1:T}) = \mathrm{N}(\mathbf{x}_k\,|\,\mathbf{m}^s_k,\mathbf{P}^s_k), \tag{4.24}$$

where

$$\begin{aligned}\mathbf{m}^s_k &= \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{m}^s_{k+1} - \mathbf{f}(\mathbf{m}_k))\\ \mathbf{P}^s_k &= \mathbf{P}_k + \mathbf{G}_k\,(\mathbf{P}^s_{k+1} - \mathbf{F}_x\,\mathbf{P}_k\,\mathbf{F}_x^T - \mathbf{Q}_k)\,\mathbf{G}_k^T.\end{aligned} \tag{4.25}$$

□

The generalization to non-additive model (3.68) is analogous to the filtering case.

4.2.2 Statistically Linearized RTS Smoother

The statistically linearized Rauch-Tung-Striebel smoother for the additive model (3.55) is the following:

Algorithm 4.2 (Statistically linearized RTS smoother)**.** *The equations for the statistically linearized RTS smoother are*

$$\begin{aligned}\mathbf{m}^-_{k+1} &= \mathrm{E}[\mathbf{f}(\mathbf{x}_k)]\\ \mathbf{P}^-_{k+1} &= \mathrm{E}[\mathbf{f}(\mathbf{x}_k)\,\delta\mathbf{x}_k^T]\,\mathbf{P}_k^{-1}\,\mathrm{E}[\mathbf{f}(\mathbf{x}_k)\,\delta\mathbf{x}_k^T]^T + \mathbf{Q}_k\\ \mathbf{G}_k &= \mathrm{E}[\mathbf{f}(\mathbf{x}_k)\,\delta\mathbf{x}_k^T]^T\,[\mathbf{P}^-_{k+1}]^{-1}\\ \mathbf{m}^s_k &= \mathbf{m}_k + \mathbf{G}_k\,[\mathbf{m}^s_{k+1} - \mathbf{m}^-_{k+1}]\\ \mathbf{P}^s_k &= \mathbf{P}_k + \mathbf{G}_k\,[\mathbf{P}^s_{k+1} - \mathbf{P}^-_{k+1}]\,\mathbf{G}_k^T,\end{aligned} \tag{4.26}$$

where the expectations are taken with respect to the filtering distribution $\mathbf{x}_k \sim \mathrm{N}(\mathbf{m}_k,\mathbf{P}_k)$.

Proof. Analogous to the ERTS case. □

The generalization to the non-additive case is also straight-forward.

4.2.3 Unscented Rauch-Tung-Striebel (URTS) Smoother

The *unscented Rauch-Tung-Striebel (URTS) smoother* (Särkkä, 2008) is a Gaussian approximation based smoother where the non-linearity is approximated using the unscented transform. The smoother equations for the additive model (3.68) are given as follows:

Algorithm 4.3 (Unscented Rauch-Tung-Striebel smoother I)**.** *The* additive form unscented RTS smoother algorithm *is the following:*

1. *Form the sigma points:*

$$\begin{aligned} \mathcal{X}_k^{(0)} &= \mathbf{m}_k, \\ \mathcal{X}_k^{(i)} &= \mathbf{m}_k + \sqrt{n+\lambda}\left[\sqrt{\mathbf{P}_k}\right]_i \\ \mathcal{X}_k^{(i+n)} &= \mathbf{m}_k - \sqrt{n+\lambda}\left[\sqrt{\mathbf{P}_k}\right]_i, \quad i = 1,\ldots,n \end{aligned} \tag{4.27}$$

   where the parameter λ *was defined in Equation* (3.99).

2. *Propagate the sigma points through the dynamic model:*

$$\hat{\mathcal{X}}_{k+1}^{(i)} = \mathbf{f}(\mathcal{X}_k^{(i)}), \quad i = 0,\ldots,2n.$$

3. *Compute the predicted mean* $\mathbf{m}_{k+1}^-$*, the predicted covariance* $\mathbf{P}_{k+1}^-$ *and the cross-covariance* $\mathbf{D}_{k+1}$*:*

$$\begin{aligned} \mathbf{m}_{k+1}^- &= \sum_{i=0}^{2n} W_i^{(m)}\, \hat{\mathcal{X}}_{k+1}^{(i)} \\ \mathbf{P}_{k+1}^- &= \sum_{i=0}^{2n} W_i^{(c)}\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T + \mathbf{Q}_k \\ \mathbf{D}_{k+1} &= \sum_{i=0}^{2n} W_i^{(c)}\,(\mathcal{X}_k^{(i)} - \mathbf{m}_k)\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T, \end{aligned} \tag{4.28}$$

   where the weights were defined in Equation (3.101).

4. *Compute the smoother gain* $\mathbf{G}_k$*, the smoothed mean* $\mathbf{m}_k^s$ *and the covariance* $\mathbf{P}_k^s$ *as follows:*

$$\begin{aligned} \mathbf{G}_k &= \mathbf{D}_{k+1}\,[\mathbf{P}_{k+1}^-]^{-1} \\ \mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \\ \mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\,(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\,\mathbf{G}_k^T. \end{aligned} \tag{4.29}$$

The above computations are started from the filtering result of the last time step $\mathbf{m}_T^s = \mathbf{m}_T$, $\mathbf{P}_T^s = \mathbf{P}_T$ *and the recursion runs backwards for* $k = T-1, \ldots, 0$.

Proof. Assume that the approximate means and covariances of the filtering distributions are available:

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) \approx \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k),$$

and the smoothing distribution of time step $k+1$ is known and approximately Gaussian

$$p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) \approx \mathrm{N}(\mathbf{x}_{k+1} \,|\, \mathbf{m}_{k+1}^s, \mathbf{P}_{k+1}^s).$$

An unscented transform based approximation to the optimal smoothing solution can be derived as follows:

1. Generate unscented transform based Gaussian approximation to the joint distribution of $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$:

$$\begin{pmatrix} \mathbf{x}_k \\ \mathbf{x}_{k+1} \end{pmatrix} \,|\, \mathbf{y}_{1:k} \sim \mathrm{N}\left(\begin{pmatrix} \mathbf{m}_k \\ \mathbf{m}_{k+1}^- \end{pmatrix}, \begin{pmatrix} \mathbf{P}_k & \mathbf{D}_{k+1} \\ \mathbf{D}_{k+1}^T & \mathbf{P}_{k+1}^- \end{pmatrix} \right), \tag{4.30}$$

   This can be done by using the additive form of the unscented transformation in Algorithm 3.13 for the nonlinearity $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k) + \mathbf{q}_k$. This is done in Equations (4.28).

2. Because the distribution (4.30) is Gaussian, by the computation rules of Gaussian distributions the conditional distribution of $\mathbf{x}_k$ is given as

$$\mathbf{x}_k \,|\, \mathbf{x}_{k+1}, \mathbf{y}_{1:T} \sim \mathrm{N}(\mathbf{m}_2, \mathbf{P}_2),$$

   where

$$\begin{aligned} \mathbf{G}_k &= \mathbf{D}_{k+1} \, [\mathbf{P}_{k+1}^-]^{-1} \\ \mathbf{m}_2 &= \mathbf{m}_k + \mathbf{G}_k (\mathbf{x}_{k+1} - \mathbf{m}_{k+1}^-) \\ \mathbf{P}_2 &= \mathbf{P}_k - \mathbf{G}_k \, \mathbf{P}_{k+1}^- \, \mathbf{G}_k^T. \end{aligned}$$

3. The rest of the derivation is completely analogous to the derivation of ERTSS in Section 4.2.1.

□

The corresponding augmented version of the smoother is almost the same, except that the augmented UT in Algorithm 3.14 is used instead of the additive UT in Algorithm 3.13. The smoother can be formulated as follows:

Algorithm 4.4 (Unscented Rauch-Tung-Striebel smoother II)**.** *A single step of the* augmented form unscented RTS smoother *is as follows:*

1. *Form the sigma points for the* $n' = n + n_q$ *-dimensional augmented random variable* $(\mathbf{x}_k^T \; \mathbf{q}_k^T)^T$

$$\begin{aligned}\tilde{\mathcal{X}}_k^{(0)} &= \tilde{\mathbf{m}}_k, \\ \tilde{\mathcal{X}}_k^{(i)} &= \tilde{\mathbf{m}}_k + \sqrt{n' + \lambda'} \left[\sqrt{\tilde{\mathbf{P}}_k}\right]_i \\ \tilde{\mathcal{X}}_k^{(i+n')} &= \tilde{\mathbf{m}}_k - \sqrt{n' + \lambda'} \left[\sqrt{\tilde{\mathbf{P}}_k}\right]_i, \quad i = 1, \ldots, n'\end{aligned} \tag{4.31}$$

where

$$\tilde{\mathbf{m}}_k = \begin{pmatrix} \mathbf{m}_k \\ \mathbf{0} \end{pmatrix} \qquad \tilde{\mathbf{P}}_k = \begin{pmatrix} \mathbf{P}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_k \end{pmatrix}.$$

2. *Propagate the sigma points through the dynamic model:*

$$\hat{\mathcal{X}}_{k+1}^{(i)} = \mathbf{f}(\tilde{\mathcal{X}}_k^{(i),x}, \tilde{\mathcal{X}}_k^{(i),q}), \quad i = 0, \ldots, 2n',$$

where $\tilde{\mathcal{X}}_k^{(i),x}$ *and* $\tilde{\mathbf{X}}_k^{(i),q}$ *denote the parts of the augmented sigma point* i*, which correspond to* $\mathbf{x}_k$ *and* $\mathbf{q}_k$*, respectively.*

3. *Compute the predicted mean* $\mathbf{m}_{k+1}^-$*, the predicted covariance* $\mathbf{P}_{k+1}^-$ *and the cross-covariance* $\mathbf{D}_{k+1}$*:*

$$\begin{aligned}\mathbf{m}_{k+1}^- &= \sum_{i=0}^{2n'} W_i^{(m)'} \, \hat{\mathcal{X}}_{k+1}^{(i)} \\ \mathbf{P}_{k+1}^- &= \sum_{i=0}^{2n'} W_i^{(c)'} \, (\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-) \, (\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T \\ \mathbf{D}_{k+1} &= \sum_{i=0}^{2n'} W_i^{(c)'} \, (\tilde{\mathcal{X}}_k^{(i),x} - \mathbf{m}_k) \, (\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T,\end{aligned} \tag{4.32}$$

where the definitions of the parameter λ' *and the weights* $W_i^{(m)'}$ *and* $W_i^{(c)'}$ *are the same as in Section 3.2.5.*

4. *Compute the smoother gain* $\mathbf{G}_k$*, the smoothed mean* $\mathbf{m}_k^s$ *and the covariance* $\mathbf{P}_k^s$*:*

$$\begin{aligned}\mathbf{G}_k &= \mathbf{D}_{k+1} \, [\mathbf{P}_{k+1}^-]^{-1} \\ \mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k \left[\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-\right] \\ \mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k \left[\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-\right] \mathbf{G}_k^T.\end{aligned} \tag{4.33}$$

4.3 General Gaussian Smoothing

4.3.1 General Gaussian Rauch-Tung-Striebel Smoother

The Gaussian moment matching described in Section 3.3.1 can be used in smoothers in analogous manner as in Gaussian assumed density filters in Section 3.3.2. If we follow the extended RTS smoother derivation in Section 4.2.1, we get the following algorithm (Särkkä and Hartikainen, 2010a):

Algorithm 4.5 (Gaussian RTS smoother I)**.** *The equations of the additive form* Gaussian RTS smoother *are the following:*

$$\begin{aligned}
\mathbf{m}_{k+1}^{-} &= \int \mathbf{f}(\mathbf{x}_k)\, \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, \mathrm{d}\mathbf{x}_k \\
\mathbf{P}_{k+1}^{-} &= \int [\mathbf{f}(\mathbf{x}_k) - \mathbf{m}_{k+1}^{-}]\, [\mathbf{f}(\mathbf{x}_k) - \mathbf{m}_{k+1}^{-}]^T \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, \mathrm{d}\mathbf{x}_k + \mathbf{Q}_k \\
\mathbf{D}_{k+1} &= \int [\mathbf{x}_k - \mathbf{m}_k]\, [\mathbf{f}(\mathbf{x}_k) - \mathbf{m}_{k+1}^{-}]^T \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, \mathrm{d}\mathbf{x}_k \\
\mathbf{G}_k &= \mathbf{D}_{k+1}\, [\mathbf{P}_{k+1}^{-}]^{-1} \\
\mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\, (\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^{-}) \\
\mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\, (\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^{-})\, \mathbf{G}_k^T.
\end{aligned} \tag{4.34}$$

The integrals above can be approximated using analogous numerical integration or analytical approximation schemes as in the filtering case, that is, with Gauss-Hermite quadratures or central differences (Ito and Xiong, 2000; Nørgaard et al., 2000; Wu et al., 2006), cubature rules (Arasaratnam and Haykin, 2009), Monte Carlo (Kotecha and Djuric, 2003), Gaussian process / Bayes-Hermite based integration (O'Hagan, 1991; Deisenroth et al., 2009), or with many other numerical integration schemes.

Algorithm 4.6 (Gaussian RTS smoother II)**.** *The equations of the non-additive form* Gaussian RTS smoother *are the following:*

$$\begin{aligned}
\mathbf{m}_{k+1}^{-} &= \int \mathbf{f}(\mathbf{x}_k, \mathbf{q}_k)\, \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, \mathrm{N}(\mathbf{q}_k \,|\, \mathbf{0}, \mathbf{Q}_k)\, \mathrm{d}\mathbf{x}_k\, \mathrm{d}\mathbf{q}_k \\
\mathbf{P}_{k+1}^{-} &= \int [\mathbf{f}(\mathbf{x}_k, \mathbf{q}_k) - \mathbf{m}_{k+1}^{-}]\, [\mathbf{f}(\mathbf{x}_k, \mathbf{q}_k) - \mathbf{m}_{k+1}^{-}]^T \\
&\qquad \times \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, \mathrm{N}(\mathbf{q}_k \,|\, \mathbf{0}, \mathbf{Q}_k)\, \mathrm{d}\mathbf{x}_k\, \mathrm{d}\mathbf{q}_k \\
\mathbf{D}_{k+1} &= \int [\mathbf{x}_k - \mathbf{m}_k]\, [\mathbf{f}(\mathbf{x}_k, \mathbf{q}_k) - \mathbf{m}_{k+1}^{-}]^T \\
&\qquad \times \mathrm{N}(\mathbf{x}_k \,|\, \mathbf{m}_k, \mathbf{P}_k)\, \mathrm{N}(\mathbf{q}_k \,|\, \mathbf{0}, \mathbf{Q}_k)\, \mathrm{d}\mathbf{x}_k\, \mathrm{d}\mathbf{q}_k \\
\mathbf{G}_k &= \mathbf{D}_{k+1}\, [\mathbf{P}_{k+1}^{-}]^{-1} \\
\mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\, (\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^{-}) \\
\mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\, (\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^{-})\, \mathbf{G}_k^T.
\end{aligned} \tag{4.35}$$

4.3.2 Gauss-Hermite Rauch-Tung-Striebel (GHRTS) Smoother

By using the Gauss-Hermite approximation to the additive form Gaussian RTS smoother, we get the following algorithm:

Algorithm 4.7 (Gauss-Hermite Rauch-Tung-Striebel smoother)**.** *The* additive form Gauss-Hermite RTS smoother algorithm *is the following:*

1. *Form the sigma points as:*

$$\mathcal{X}_k^{(i_1,\ldots,i_n)} = \mathbf{m}_k + \sqrt{\mathbf{P}_k}\,\boldsymbol{\xi}^{(i_1,\ldots,i_n)} \qquad i_1,\ldots,i_n = 1,\ldots,p, \tag{4.36}$$

   where the unit sigma points $\boldsymbol{\xi}^{(i_1,\ldots,i_n)}$ *were defined in Equation* (3.169).

2. *Propagate the sigma points through the dynamic model:*

$$\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} = \mathbf{f}(\mathcal{X}_k^{(i_1,\ldots,i_n)}), \quad i_1,\ldots,i_n = 1,\ldots,p, \tag{4.37}$$

3. *Compute the predicted mean* $\mathbf{m}_{k+1}^-$*, the predicted covariance* $\mathbf{P}_{k+1}^-$ *and the cross-covariance* $\mathbf{D}_{k+1}$*:*

$$\begin{aligned}
\mathbf{m}_{k+1}^- &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)}\,\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} \\
\mathbf{P}_{k+1}^- &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)}(\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} - \mathbf{m}_{k+1}^-)\,(\hat{\mathcal{X}}_{k+1}^{(i_1,\ldots,i_n)} - \mathbf{m}_{k+1}^-)^T + \mathbf{Q}_k \\
\mathbf{D}_{k+1} &= \sum_{i_1,\ldots,i_n} W^{(i_1,\ldots,i_n)}(\mathcal{X}_k^{(i)} - \mathbf{m}_k)\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T,
\end{aligned} \tag{4.38}$$

   where the weights $W^{(i_1,\ldots,i_n)}$ *were defined in Equation* (3.168).

4. *Compute the gain* $\mathbf{G}_k$*, mean* $\mathbf{m}_k^s$ *and covariance* $\mathbf{P}_k^s$ *as follows:*

$$\begin{aligned}
\mathbf{G}_k &= \mathbf{D}_{k+1}\,[\mathbf{P}_{k+1}^-]^{-1} \\
\mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \\
\mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\,(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\,\mathbf{G}_k^T.
\end{aligned} \tag{4.39}$$

4.3.3 Cubature Rauch-Tung-Striebel (CRTS) Smoother

By using the 3rd order spherical cubature approximation to the additive form Gaussian RTS smoother, we get the following algorithm:

Algorithm 4.8 (Cubature Rauch-Tung-Striebel smoother I)**.** *The* additive form cubature RTS smoother algorithm *is the following:*

1. *Form the sigma points:*
$$\mathcal{X}_k^{(i)} = \mathbf{m}_k + \sqrt{\mathbf{P}_k}\,\boldsymbol{\xi}^{(i)}, \qquad i = 1,\ldots,2n, \tag{4.40}$$
where the unit sigma points are defined as
$$\boldsymbol{\xi}^{(i)} = \begin{cases} \sqrt{n}\,\mathbf{e}_i & , \quad i = 1,\ldots,n \\ -\sqrt{n}\,\mathbf{e}_{i-n} & , \quad i = n+1,\ldots,2n. \end{cases} \tag{4.41}$$
2. *Propagate the sigma points through the dynamic model:*
$$\hat{\mathcal{X}}_{k+1}^{(i)} = \mathbf{f}(\mathcal{X}_k^{(i)}), \quad i = 1,\ldots,2n.$$
3. *Compute the predicted mean $\mathbf{m}_{k+1}^-$, the predicted covariance $\mathbf{P}_{k+1}^-$ and the cross-covariance $\mathbf{D}_{k+1}$:*
$$\begin{aligned} \mathbf{m}_{k+1}^- &= \frac{1}{2n}\sum_{i=1}^{2n} \hat{\mathcal{X}}_{k+1}^{(i)} \\ \mathbf{P}_{k+1}^- &= \frac{1}{2n}\sum_{i=1}^{2n} (\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T + \mathbf{Q}_k \\ \mathbf{D}_{k+1} &= \frac{1}{2n}\sum_{i=1}^{2n} (\mathcal{X}_k^{(i)} - \mathbf{m}_k)\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T. \end{aligned} \tag{4.42}$$
4. *Compute the gain $\mathbf{G}_k$, mean $\mathbf{m}_k^s$ and covariance $\mathbf{P}_k^s$ as follows:*
$$\begin{aligned} \mathbf{G}_k &= \mathbf{D}_{k+1}\,[\mathbf{P}_{k+1}^-]^{-1} \\ \mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\,(\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \\ \mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\,(\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-)\,\mathbf{G}_k^T. \end{aligned} \tag{4.43}$$

By using the 3rd order spherical cubature approximation to the non-additive form Gaussian RTS smoother, we get the following algorithm:

Algorithm 4.9 (Cubature Rauch-Tung-Striebel smoother II)**.** *A single step of the* augmented form cubature RTS smoother *is as follows:*

1. *Form the sigma points for the $n' = n + n_q$ -dimensional augmented random variable $(\mathbf{x}_k^T\ \mathbf{q}_k^T)^T$*
$$\tilde{\mathcal{X}}_k^{(i)} = \tilde{\mathbf{m}}_k + \sqrt{\tilde{\mathbf{P}}_k}\,\boldsymbol{\xi}^{(i)'} \qquad i = 1,\ldots,2n', \tag{4.44}$$
where
$$\tilde{\mathbf{m}}_k = \begin{pmatrix} \mathbf{m}_k \\ \mathbf{0} \end{pmatrix} \qquad \tilde{\mathbf{P}}_k = \begin{pmatrix} \mathbf{P}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_k \end{pmatrix}.$$

2. *Propagate the sigma points through the dynamic model:*

$$\hat{\mathcal{X}}_{k+1}^{(i)} = \mathbf{f}(\tilde{\mathcal{X}}_k^{(i),x}, \tilde{\mathcal{X}}_k^{(i),q}), \quad i = 1, \ldots, 2n',$$

where $\tilde{\mathcal{X}}_k^{(i),x}$ *and* $\tilde{\mathbf{X}}_k^{(i),q}$ *denote the parts of the augmented sigma point* i*, which correspond to* $\mathbf{x}_k$ *and* $\mathbf{q}_k$*, respectively.*

3. *Compute the predicted mean* $\mathbf{m}_{k+1}^-$*, the predicted covariance* $\mathbf{P}_{k+1}^-$ *and the cross-covariance* $\mathbf{D}_{k+1}$*:*

$$\begin{aligned}
\mathbf{m}_{k+1}^- &= \frac{1}{2n'} \sum_{i=1}^{2n'} \hat{\mathcal{X}}_{k+1}^{(i)} \\
\mathbf{P}_{k+1}^- &= \frac{1}{2n'} \sum_{i=1}^{2n'} (\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T \\
\mathbf{D}_{k+1} &= \frac{1}{2n'} \sum_{i=1}^{2n'} (\tilde{\mathcal{X}}_k^{(i),x} - \mathbf{m}_k)\,(\hat{\mathcal{X}}_{k+1}^{(i)} - \mathbf{m}_{k+1}^-)^T.
\end{aligned} \tag{4.45}$$

4. *Compute the gain* $\mathbf{G}_k$*, mean* $\mathbf{m}_k^s$ *and covariance* $\mathbf{P}_k^s$*:*

$$\begin{aligned}
\mathbf{G}_k &= \mathbf{D}_{k+1}\,[\mathbf{P}_{k+1}^-]^{-1} \\
\mathbf{m}_k^s &= \mathbf{m}_k + \mathbf{G}_k\,\left[\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-\right] \\
\mathbf{P}_k^s &= \mathbf{P}_k + \mathbf{G}_k\,\left[\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-\right]\,\mathbf{G}_k^T.
\end{aligned} \tag{4.46}$$

4.4 Fixed-Point and Fixed-Lag Gaussian Smoothing

The smoother algorithms that we have considered this far have all been *fixed-interval* smoothing algorithms, which can be used for computing estimates of a fixed time interval of states given the measurements on the same interval. However, there exists a couple of other types of smoothing problems as well:

- *Fixed-point smoothing* refers to a methodology which can be used for efficiently computing the optimal estimate of the *initial state* or some other fixed-time state of a state space model, given an increasing number of measurements after it. This kind of estimation problem arises, for example, in estimation of the state of a spacecraft at a given point of time in the past (Meditch, 1969) or in alignment and calibration of inertial navigation systems (Grewal et al., 1988).

- *Fixed-lag smoothing* is a methodology for computing delayed estimates of state space models given measurements up to the current time and a constant horizon in the future. Fixed-lag smoothing can be considered as optimal filtering, where a constant delay is tolerated in the estimates. Potential applications of fixed-lag smoothing are all the problems where optimal filters are typically applied, and where the small delay is tolerated. An example of such application is digital demodulation (Tam et al., 1973).

The presentation here is based on the results presented in (Särkkä and Hartikainen, 2010a), except that the derivations are presented in a bit more detail than in the original reference.

4.4.1 General Fixed-Point Smoother Equations

The general fixed-interval RTS smoothers described in this document have the property that given the gain sequence, we only need linear operations for performing the smoothing, and in this sense, the smoothing is a completely *linear operation*. The only non-linear operations in the smoother are in the approximations of the Gaussian integrals. However, these operations are performed to the filtering results and thus we can compute the smoothing gain sequence $\mathbf{G}_k$ from the filtering results in a causal manner. Because of these properties we may now derive a fixed-point smoother using similar methods as have been used for deriving the linear fixed-point smoother from the linear Rauch-Tung-Striebel smoother in (Meditch, 1969).

We shall now denote the smoothing means and covariances using notation of type $\mathbf{m}_{k|n}$ and $\mathbf{P}_{k|n}$, which refer to the mean and covariance of the state $\mathbf{x}_k$, which is conditioned to measurement $\mathbf{y}_1, \ldots, \mathbf{y}_n$. With this notation, the filter estimates are $\mathbf{m}_{k|k}$, $\mathbf{P}_{k|k}$ and the RTS smoother estimates, which are conditioned to T measurements have the form $\mathbf{m}_{k|T}$, $\mathbf{P}_{k|T}$. The RTS smoothers have the following common recursion equations:

$$\begin{aligned} \mathbf{G}_k &= \mathbf{D}_{k+1} \, [\mathbf{P}_{k+1}^-]^{-1} \\ \mathbf{m}_{k|T} &= \mathbf{m}_k + \mathbf{G}_k \, \left[\mathbf{m}_{k+1|T} - \mathbf{m}_{k+1}^-\right] \\ \mathbf{P}_{k|T} &= \mathbf{P}_k + \mathbf{G}_k \, \left[\mathbf{P}_{k+1|T} - \mathbf{P}_{k+1}^-\right] \, \mathbf{G}_k^T. \end{aligned} \tag{4.47}$$

which are indeed linear recursion equations for the smoother mean and covariance. Note that the gains $\mathbf{G}_k$ only depend on the filtering results, not on the smoother mean and covariance. Because the gains $\mathbf{G}_k$ are independent of T, from the equations (4.47) we get for $i = j, \ldots, k$ the identity:

$$\mathbf{m}_{i|k} - \mathbf{m}_{i|i} = \mathbf{G}_i [\mathbf{m}_{i+1|k} - \mathbf{m}_{i+1|i}]. \tag{4.48}$$

Similarly, for $i = j, \ldots, k-1$ we have

$$\mathbf{m}_{i|k-1} - \mathbf{m}_{i|i} = \mathbf{G}_i [\mathbf{m}_{i+1|k-1} - \mathbf{m}_{i+1|i}]. \tag{4.49}$$

Subtracting these equations gives the identity

$$\mathbf{m}_{i|k} - \mathbf{m}_{i|k-1} = \mathbf{G}_i [\mathbf{m}_{i+1|k} - \mathbf{m}_{i+1|k-1}]. \tag{4.50}$$

By varying i from j to $k-1$ we get the identities

$$\begin{aligned}
\mathbf{m}_{j|k} - \mathbf{m}_{j|k-1} &= \mathbf{G}_j[\mathbf{m}_{j+1|k} - \mathbf{m}_{j+1|k-1}] \\
\mathbf{m}_{j+1|k} - \mathbf{m}_{j+1|k-1} &= \mathbf{G}_{j+1}[\mathbf{m}_{j+2|k} - \mathbf{m}_{j+2|k-1}] \\
&\vdots \\
\mathbf{m}_{k-1|k} - \mathbf{m}_{k-1|k-1} &= \mathbf{G}_{k-1}[\mathbf{m}_{k|k} - \mathbf{m}_{k|k-1}].
\end{aligned} \tag{4.51}$$

If we sequentially substitute the above equations to each other starting from the last and proceeding to the first, we get the equation

$$\mathbf{m}_{j|k} = \mathbf{m}_{j|k-1} + \mathbf{B}_{j|k}[\mathbf{m}_{k|k} - \mathbf{m}_{k|k-1}], \tag{4.52}$$

where

$$\mathbf{B}_{j|k} = \mathbf{G}_j \times \cdots \times \mathbf{G}_{k-1}. \tag{4.53}$$

Analogously for the covariance we get

$$\mathbf{P}_{j|k} = \mathbf{P}_{j|k-1} + \mathbf{B}_{j|k}[\mathbf{P}_{k|k} - \mathbf{P}_{k|k-1}]\mathbf{B}_{j|k}^T. \tag{4.54}$$

The *general fixed-point smoother* algorithm for smoothing the time point j can now be implemented by performing the following operations at each time step $k = 1, 2, 3, \ldots$:

1. *Gain computation:* Compute the predicted mean $\mathbf{m}_{k|k-1}$, predicted covariance $\mathbf{P}_{k|k-1}$ and cross-covariance $\mathbf{D}_k$ from the filtering results using one of equations in the smoother algorithms. Then compute the gain from the equation
$$\mathbf{G}_{k-1} = \mathbf{D}_k\,[\mathbf{P}_k^-]^{-1}. \tag{4.55}$$

2. *Fixed-point smoothing:*

   (a) If $k < j$, just store the filtering result.

   (b) If $k = j$, set $\mathbf{B}_{j|j} = \mathbf{I}$. The fixed-point smoothed mean and covariance on step j are equal to the filtered mean and covariance $\mathbf{m}_{j|j}$ and $\mathbf{P}_{j|j}$.

   (c) If $k > j$, compute the smoothing gain and the fixed-point smoother mean and covariance:
$$\begin{aligned}
\mathbf{B}_{j|k} &= \mathbf{B}_{j|k-1}\mathbf{G}_{k-1} \\
\mathbf{m}_{j|k} &= \mathbf{m}_{j|k-1} + \mathbf{B}_{j|k}[\mathbf{m}_{k|k} - \mathbf{m}_{k|k-1}] \\
\mathbf{P}_{j|k} &= \mathbf{P}_{j|k-1} + \mathbf{B}_{j|k}[\mathbf{P}_{k|k} - \mathbf{P}_{k|k-1}]\mathbf{B}_{j|k}^T.
\end{aligned} \tag{4.56}$$

Because only a constant number of computations is needed on each time step, the algorithm can be easily implemented in real time.

4.4.2 General Fixed-Lag Smoother Equations

It is also possible to derive a general fixed-lag smoother by using a similar procedure as in the previous section. However, this approach will lead to a numerically unstable algorithm as will be seen shortly. Let n be the number of lags. From the fixed-point smoother we get

$$\begin{aligned}\mathbf{m}_{k-n-1|k} &= \mathbf{m}_{k-n-1|k-1} \\ &\quad + \mathbf{B}_{k-n-1|k}[\mathbf{m}_{k|k} - \mathbf{m}_{k|k-1}].\end{aligned} \tag{4.57}$$

From the fixed-interval smoother we get

$$\begin{aligned}\mathbf{m}_{k-n-1|k} &= \mathbf{m}_{k-n-1|k-n-1} \\ &\quad + \mathbf{G}_{k-1-n}[\mathbf{m}_{k-n|k} - \mathbf{m}_{k-n|k-n-1}].\end{aligned} \tag{4.58}$$

Equating the right hand sides, and solving for $\mathbf{m}_{k-n|k}$ while remembering the identity $\mathbf{B}_{k-n|k} = \mathbf{G}_{k-n-1}^{-1}\mathbf{B}_{k-n-1|k}$ results in the smoothing equation

$$\begin{aligned}\mathbf{m}_{k-n|k} &= \mathbf{m}_{k-n|k-n-1} \\ &\quad + \mathbf{G}_{k-n-1}^{-1}[\mathbf{m}_{k-n-1|k-1} - \mathbf{m}_{k-n-1|k-n-1}] \\ &\quad + \mathbf{B}_{k-n|k}[\mathbf{m}_{k|k} - \mathbf{m}_{k|k-1}].\end{aligned} \tag{4.59}$$

Similarly, for covariance we get

$$\begin{aligned}\mathbf{P}_{k-n|k} &= \mathbf{P}_{k-n|k-n-1} \\ &\quad + \mathbf{G}_{k-n-1}^{-1}[\mathbf{P}_{k-n-1|k-1} - \mathbf{P}_{k-n-1|k-n-1}]\mathbf{G}_{k-n-1}^{-T} \\ &\quad + \mathbf{B}_{k-n|k}[\mathbf{P}_{k|k} - \mathbf{P}_{k|k-1}]\mathbf{B}_{k-n|k}^{T}.\end{aligned} \tag{4.60}$$

The equations (4.59) and (4.60) can be, *in principle*, used for recursively computing the fixed-lag smoothing solution. The number of computations does not depend on the lag length. This solution can be seen to be of the same form as the fixed-lag smoother given in (Rauch, 1963; Meditch, 1969; Crassidis and Junkins, 2004). Unfortunately, it has been shown (Kelly and Anderson, 1971) that this form of smoother is *numerically unstable* and thus not usable in practice. However, sometimes the equations do indeed work and can be used if the user is willing to take the risk of potential instability.

In (Moore, 1973; Moore and Tam, 1973) stable algorithms for optimal fixed-lag smoothing are derived by augmenting the n lagged states to a Kalman filter. This approach ensures the stability of the algorithm. Using certain simplifications it is possible to reduce the computations, and this is also possible when certain types of extended Kalman filters are used (Moore, 1973; Moore and Tam, 1973). Unfortunately, such simplifications cannot be done in more general case and, for example, when the unscented transformation (Julier et al., 1995, 2000) or a quadrature rule (Ito and Xiong, 2000) is used, the required amount of computations becomes high, because the Cholesky factorization of the whole joint covariance of the n lagged states would be needed in the computations. Another possibility, which is employed here, is to take advantage of the fact that Rauch-Tung-Striebel smoother equations are numerically stable and can be used for fixed-lag smoothing. The fixed-lag smoothing can be efficiently implemented by taking into account that the gain sequence needs to be evaluated only once, and the same gains can be used in different smoothers operating on different intervals. Thus the *general fixed-lag smoother* can be implemented by performing the following on each time step $k = 1, 2, 3, \ldots$:

1. *Gain computation:* During the Gaussian filter prediction step compute and store the predicted mean $\mathbf{m}_{k|k-1}$, predicted covariance $\mathbf{P}_{k|k-1}$ and cross-covariance $\mathbf{D}_k$ using one of equations in the smoother algorithms. Also compute and store the smoothing gain

$$\mathbf{G}_{k-1} = \mathbf{D}_k \, [\mathbf{P}_k^-]^{-1}. \tag{4.61}$$

2. *Fixed-lag smoothing:* Using the stored gain sequence, compute the smoothing solutions for steps $j = k - n, \ldots, k$ using the following backward recursion, starting from the filtering solution on step $j = k$:

$$\begin{aligned} \mathbf{m}_{j|k} &= \mathbf{m}_{j|j} + \mathbf{G}_j \, \left[\mathbf{m}_{j+1|k} - \mathbf{m}_{j+1|j}\right] \\ \mathbf{P}_{j|k} &= \mathbf{P}_{j|j} + \mathbf{G}_j \, \left[\mathbf{P}_{j+1|k} - \mathbf{P}_{j+1|j}\right] \, \mathbf{G}_j^T. \end{aligned} \tag{4.62}$$

The required number of computations per time step grows linearly with the length of lag. Thus the computational requirements are comparable to algorithms presented in (Moore, 1973; Moore and Tam, 1973). The algorithm defined in equations (4.59) and (4.60) would be computationally more efficient, but as already stated, it would be numerically unstable.

4.5 Monte Carlo Based Smoothers

4.5.1 SIR Particle Smoother

The *SIR particle smoother* of Kitagawa (1996) is based on direct usage of SIR for smoothing. Recall that in Section 3.4.3 we derived the sequential importance sampling (SIS) method to approximate the full posterior distribution, not just the filtering distributions. We then discarded the sample histories $\mathbf{x}_{1:k}^{(i)}$ and only kept the current states $\mathbf{x}_k^{(i)}$, because we were interested in the filtering distributions. But we can get an approximation to the smoothing distribution by keeping the full histories. To get the smoothing solution from sequential importance resampling (SIR) we also need to resample the state histories, not only the current states, to prevent the resampling from breaking the state histories. The resulting algorithm is the following:

Algorithm 4.10 (SIR particle smoother)**.** *The direct sequential importance resampling (SIR) based smoother algorithm is the following:*

- *Draw N samples $\mathbf{x}_0^{(i)}$ from the prior*

$$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \qquad i = 1, \ldots, N, \tag{4.63}$$

  and set $w_0^{(i)} = 1/N$, for all $i = 1, \ldots, N$. Initialize the state histories to contain the prior samples $\mathbf{x}_0^{(i)}$.

- *For each $k = 1, \ldots, T$ do the following:*

  1. *Draw N new samples $\mathbf{x}_k^{(i)}$ from the importance distributions*

$$\mathbf{x}_k^{(i)} \sim \pi(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_{1:k}), \qquad i = 1, \ldots, N, \tag{4.64}$$

     where $\mathbf{x}_{k-1}^{(i)}$ is the $k-1$th (last) element in the sample history $\mathbf{x}_{1:k-1}^{(i)}$.
  2. *Calculate the new weights according to*

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k \,|\, \mathbf{x}_k^{(i)})\, p(\mathbf{x}_k^{(i)} \,|\, \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} \,|\, \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_{1:k})} \tag{4.65}$$

     and normalize them to sum to unity.
  3. *Append the samples to the state histories:*

$$\mathbf{x}_{1:k}^{(i)} = (\mathbf{x}_{1:k-1}^{(i)}, \mathbf{x}_k^{(i)}). \tag{4.66}$$

  4. *If the effective number of particles* (3.234) *is too low, perform resampling to the state histories.*

The approximation to the full posterior (smoothed) distribution is (Kitagawa, 1996; Doucet et al., 2000):

$$p(\mathbf{x}_{1:T} \,|\, \mathbf{y}_{1:T}) \approx \sum_{i=1}^{N} w_T^{(i)} \delta(\mathbf{x}_{1:T} - \mathbf{x}_{1:T}^{(i)}). \tag{4.67}$$

The approximation to the smoothed posterior distribution at time step k, given the measurements up to the time step $T > k$ is

$$p(\mathbf{x}_k \,|\, \mathbf{y}_{1:T}) \approx \sum_{i=1}^{N} w_T^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}), \tag{4.68}$$

where $\mathbf{x}_k^{(i)}$ is the kth component in $\mathbf{x}_{1:T}^{(i)}$. However, if $T \gg k$ the direct SIR smoother algorithm is known to produce very degenerate approximations (Kitagawa, 1996; Doucet et al., 2000).

4.5.2 Backward-Simulation Particle Smoother

A less degenerate particle smoother than the SIR particle smoother can be obtained by reusing the filtering results instead of simply storing the full particle histories in SIR. The *backward-simulation particle smoother* of Godsill et al. (2004) is based on simulation of individual trajectories backwards, starting from the last step and proceeding to the first.

Assume that we have already simulated a trajectory $\tilde{\mathbf{x}}_{k+1:T}$ from the smoothing distribution. By using the Equation (4.3) we get:

$$\begin{aligned} p(\mathbf{x}_k \mid \tilde{\mathbf{x}}_{k+1}, \mathbf{y}_{1:T}) &= \frac{p(\tilde{\mathbf{x}}_{k+1} \mid \mathbf{x}_k)\, p(\tilde{\mathbf{x}}_k \mid \mathbf{y}_{1:k})}{p(\tilde{\mathbf{x}}_{k+1} \mid \mathbf{y}_{1:k})} \\ &= Z\, p(\tilde{\mathbf{x}}_{k+1} \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{y}_{1:k}), \end{aligned} \tag{4.69}$$

where Z is a normalization constant. By substituting the SIR filter approximation we get

$$p(\mathbf{x}_k \mid \tilde{\mathbf{x}}_{k+1}, \mathbf{y}_{1:T}) \approx Z \sum_i w_k^{(i)}\, p(\tilde{\mathbf{x}}_{k+1} \mid \mathbf{x}_k^{(i)})\, \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}). \tag{4.70}$$

We can now draw a sample from this distribution by sampling $\mathbf{x}_k^{(i)}$ with probability $\propto w_k^{(i)} p(\tilde{\mathbf{x}}_{k+1} \mid \mathbf{x}_k^{(i)})$. The resulting algorithm is the following:

Algorithm 4.11 (Backward-simulation particle smoother)**.** *Given the weighted set of particles* $\{w_k^{(i)}, \mathbf{x}_k^{(i)} \mid i = 1, \ldots, N,\ k = 1, \ldots, T\}$ *representing the filtering distributions:*

- *Choose* $\tilde{\mathbf{x}}_T = \mathbf{x}_T^{(i)}$ *with probability* $w_T^{(i)}$.
- *For* $k = T-1, \ldots, 0$*:*
  1. *Compute new weights by*
  $$w_{k|k+1}^{(i)} \propto w_k^{(i)} p(\tilde{\mathbf{x}}_{k+1} \mid \mathbf{x}_k^{(i)}). \tag{4.71}$$
  2. *Choose* $\tilde{\mathbf{x}}_k = \mathbf{x}_k^{(i)}$ *with probability* $w_{k|k+1}^{(i)}$.

Given S iterations of the above procedure resulting in sample trajectories $\tilde{\mathbf{x}}_{1:T}^{(j)}$ for $j = 1, \ldots, S$ the smoothing distribution can now be approximated as

$$p(\mathbf{x}_{1:T} \mid \mathbf{y}_{1:T}) \approx \frac{1}{S} \sum_j \delta(\mathbf{x}_{1:T} - \tilde{\mathbf{x}}_{1:T}^{(j)}). \tag{4.72}$$

The marginal distribution samples for a step k can be obtained by extracting the kth components from the above trajectories. The computational complexity of the method is $O(S\,T\,N)$. However, the result is much less degenerate than of the particle smoother of Kitagawa (1996).

4.5.3 Reweighting Particle Smoother

The reweighting particle smoother of Hürzeler and Kunsch (1998) and Doucet et al. (2000) is based on computing new weights for the SIR filtering particles such that we get an approximation to the smoothing distribution. Assume that have already computed the weights for the following approximation, where the particles $\mathbf{x}_{k+1}^{(i)}$ are from the SIR filter:

$$p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) \approx \sum_i w_{k+1|T}^{(i)} \, \delta(\mathbf{x}_{k+1} - \mathbf{x}_{k+1}^{(i)}). \tag{4.73}$$

The integral in the second of Equations (4.2) can be now approximated as:

$$\begin{aligned}
&\int \frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k) \, p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T})}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})} \, \mathrm{d}\mathbf{x}_{k+1} \\
&\quad \approx \int \frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k)}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})} \sum_i \left[w_{k+1|T}^{(i)} \, \delta(\mathbf{x}_{k+1} - \mathbf{x}_{k+1}^{(i)}) \right] \mathrm{d}\mathbf{x}_{k+1} \\
&\quad = \sum_i w_{k+1|T}^{(i)} \, \frac{p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{x}_k)}{p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{y}_{1:k})}.
\end{aligned} \tag{4.74}$$

By using SIR filter approximation we get the following approximation for the predicted distribution in the denominator:

$$p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k}) \approx \sum_j w_k^{(j)} \, p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k^{(j)}), \tag{4.75}$$

which gives

$$\int \frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T}) \, p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k)}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})} \, \mathrm{d}\mathbf{x}_{k+1} \approx \sum_i w_{k+1|T}^{(i)} \, \frac{p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{x}_k)}{\left[\sum_j w_k^{(j)} \, p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{x}_k^{(j)}) \right]}. \tag{4.76}$$

By substituting the SIR filter approximation and the approximation above to the Bayesian optimal smoothing equation we get:

$$\begin{aligned}
p(\mathbf{x}_k \,|\, \mathbf{y}_{1:T}) &= p(\mathbf{x}_k \,|\, \mathbf{y}_{1:k}) \int \left[\frac{p(\mathbf{x}_{k+1} \,|\, \mathbf{x}_k) \, p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:T})}{p(\mathbf{x}_{k+1} \,|\, \mathbf{y}_{1:k})} \right] \mathrm{d}\mathbf{x}_{k+1} \\
&\approx \sum_l w_k^{(l)} \, \delta(\mathbf{x}_k - \mathbf{x}_k^{(l)}) \sum_i w_{k+1|T}^{(i)} \, \frac{p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{x}_k^{(l)})}{\left[\sum_j w_k^{(j)} \, p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{x}_k^{(j)}) \right]},
\end{aligned} \tag{4.77}$$

where we can identify the weights as

$$w_{k|T}^{(l)} \propto \sum_i w_{k+1|T}^{(i)} \, \frac{w_k^{(l)} p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{x}_k^{(l)})}{\left[\sum_j w_k^{(j)} \, p(\mathbf{x}_{k+1}^{(i)} \,|\, \mathbf{x}_k^{(j)}) \right]}. \tag{4.78}$$

Thus we get the following algorithm:

Algorithm 4.12 (Reweighting particle smoother)**.** *Given the weighted set of particles* $\{w_k^{(i)}, \mathbf{x}_k^{(i)} \mid i = 1, \ldots, N\}$ *representing the filtering distribution, we can form approximations to the marginal smoothing distributions as follows:*

- *Start by setting* $w_{T|T}^{(i)} = w_T^{(i)}$ *for* $i = 1, \ldots, N$.
- *For each* $k = T-1, \ldots, 0$ *do the following:*
  - *Compute new importance weights by*

$$w_{k|T}^{(i)} \propto \sum_j w_{k+1|T}^{(j)} \frac{w_k^{(i)} p(\mathbf{x}_{k+1}^{(j)} \mid \mathbf{x}_k^{(i)})}{\left[\sum_l w_k^{(l)} \, p(\mathbf{x}_{k+1}^{(j)} \mid \mathbf{x}_k^{(l)})\right]}. \tag{4.79}$$

- *At each step* k *the marginal smoothing distribution can be approximated as*

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) \approx \sum_i w_{k|T}^{(i)} \, \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}). \tag{4.80}$$

The computational complexity of the method is $O(T\,N^2)$, that is, the same as of the backward-simulation smoother with $S = N$ simulated trajectories.

4.5.4 Rao-Blackwellized Particle Smoothers

Rao-Blackwellized particle smoothers (RFPS) can be used for computing approximate smoothing solutions to conditionally Gaussian models defined in Equation (3.241). A simple way to implement a RBPS is to store the histories instead of the single states in RBPF, as in the case of SIR particle smoother (Algorithm 4.10). The corresponding histories of the means and the covariances are then conditional on the *latent variable histories* $\boldsymbol{\theta}_{1:T}$. However, the means and covariances at time step k are only conditional on the *measurement histories* up to k, not on the later measurements. In order to correct this, RTS smoothers have to be applied to each history of the means and the covariances:

Algorithm 4.13 (Rao-Blackwellized SIR particle smoother)**.** *A set of weighted samples* $\{w_T^{s,(i)}, \boldsymbol{\theta}_{1:T}^{s,(i)}, \mathbf{m}_{1:T}^{s,(i)}, \mathbf{P}_{1:T}^{s,(i)} \;:\; i = 1, \ldots, N\}$ *representing the smoothed distribution can be computed as follows:*

1. *Compute the weighted set of Rao-Blackwellized state histories*

$$\{w_T^{(i)}, \boldsymbol{\theta}_{1:T}^{(i)}, \mathbf{m}_{1:T}^{(i)}, \mathbf{P}_{1:T}^{(i)} \;:\; i = 1, \ldots, N\} \tag{4.81}$$

   by storing histories in the Rao-Blackwellized particle filter analogously to the SIR particle smoother in Algorithm 4.10.

2. *Set*

$$\begin{aligned} w_T^{s,(i)} &= w_T^{(i)} \\ \boldsymbol{\theta}_{1:T}^{s,(i)} &= \boldsymbol{\theta}_{1:T}^{(i)}. \end{aligned} \tag{4.82}$$

3. *Apply the RTS smoother to each of the mean and covariance histories* $\mathbf{m}_{1:T}^{(i)}, \mathbf{P}_{1:T}^{(i)}$ *for* $i = 1, \ldots, N$ *to produce the smoothed mean and covariance histories* $\mathbf{m}_{1:T}^{s,(i)}, \mathbf{P}_{1:T}^{s,(i)}$.

The Rao-Blackwellized particle smoother in this simple form also has the same disadvantage as the SIR particle smoother, that is, the smoothed estimate of $\boldsymbol{\theta}_k$ can be quite degenerate if $T \gg k$. Fortunately, the smoothed estimates of the actual states $\mathbf{x}_k$ can still be relatively good, because their degeneracy is avoided by Rao-Blackwellization.

To avoid the degeneracy in estimates of $\boldsymbol{\theta}_k$ it is possible to use better sampling procedures for generating samples from the smoothing distributions analogously to the plain particle smoothing. The backward-simulation has indeed been generalized to Rao-Blackwellized case, but the implementation of the Rao-Blackwellized reweighting smoother seems to be quite problematic.

The Rao-Blackwellized backward-simulation smoother proposed by Särkkä et al. (2012a) can be used for drawing backward trajectories from the marginal posterior of the latent variables $\boldsymbol{\theta}_k$ and posterior of the conditionally Gaussian part is obtained via Kalman filtering and RTS smoothing. The methods of Fong et al. (2002) and Lindsten and Schön (2011) are based on simulating backward trajectories from the joint distribution $(\mathbf{x}_k, \boldsymbol{\theta}_k)$. However, these smoothers are not really Rao-Blackwellized backward-simulation smoothers, because they require sampling of the linear part of the state as well. It is also possible to construct approximate (Kim's approximation based) backward-simulation smoothers by replacing transition density $p(\mathbf{x}_{k+1} \mid \mathbf{x}_k)$ in the Algorithm 4.11 with $p(\boldsymbol{\theta}_{k+1} \mid \boldsymbol{\theta}_k)$ (see Särkkä et al., 2012a). Given a trajectory of the non-Gaussian variable, the linear Gaussian part may be recovered with Kalman filter and RTS smoother.

Appendix A

Additional Material

A.1 Properties of Gaussian Distribution

Definition A.1 (Gaussian distribution)**.** *Random variable* $\mathbf{x} \in \mathbb{R}^n$ *has Gaussian distribution with mean* $\mathbf{m} \in \mathbb{R}^n$ *and covariance* $\mathbf{P} \in \mathbb{R}^{n \times n}$ *its probability density has the form*

$$\mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) = \frac{1}{(2\,\pi)^{n/2}\, |\mathbf{P}|^{1/2}} \exp\left(-\frac{1}{2} (\mathbf{x} - \mathbf{m})^T\, \mathbf{P}^{-1}\, (\mathbf{x} - \mathbf{m}) \right), \tag{A.1}$$

where $|\mathbf{P}|$ *is the determinant of matrix* $\mathbf{P}$.

Lemma A.1 (Joint density of Gaussian variables)**.** *If random variables* $\mathbf{x} \in \mathbb{R}^n$ *and* $\mathbf{y} \in \mathbb{R}^m$ *have the Gaussian probability densities*

$$\begin{aligned} \mathbf{x} &\sim \mathrm{N}(\mathbf{x} \,|\, \mathbf{m}, \mathbf{P}) \\ \mathbf{y} \,|\, \mathbf{x} &\sim \mathrm{N}(\mathbf{y} \,|\mathbf{H}\, \mathbf{x} + \mathbf{u}, \mathbf{R}), \end{aligned} \tag{A.2}$$

then the joint density of $\mathbf{x}, \mathbf{y}$ *and the marginal distribution of* $\mathbf{y}$ *are given as*

$$\begin{aligned} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} &\sim \mathrm{N}\left(\begin{bmatrix} \mathbf{m} \\ \mathbf{H}\, \mathbf{m} + \mathbf{u} \end{bmatrix}, \begin{bmatrix} \mathbf{P} & \mathbf{P}\, \mathbf{H}^T \\ \mathbf{H}\, \mathbf{P} & \mathbf{H}\, \mathbf{P}\, \mathbf{H}^T + \mathbf{R} \end{bmatrix} \right) \\ \mathbf{y} &\sim \mathrm{N}(\mathbf{H}\, \mathbf{m} + \mathbf{u}, \mathbf{H}\, \mathbf{P}\, \mathbf{H}^T + \mathbf{R}). \end{aligned} \tag{A.3}$$

Lemma A.2 (Conditional density of Gaussian variables)**.** *If the random variables* $\mathbf{x}$ *and* $\mathbf{y}$ *have the joint Gaussian probability density*

$$\mathbf{x}, \mathbf{y} \sim \mathrm{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^T & \mathbf{B} \end{bmatrix} \right), \tag{A.4}$$

then the marginal and conditional densities of $\mathbf{x}$ *and* $\mathbf{y}$ *are given as follows:*

$$\begin{aligned} \mathbf{x} &\sim \mathrm{N}(\mathbf{a}, \mathbf{A}) \\ \mathbf{y} &\sim \mathrm{N}(\mathbf{b}, \mathbf{B}) \\ \mathbf{x} \,|\, \mathbf{y} &\sim \mathrm{N}(\mathbf{a} + \mathbf{C}\, \mathbf{B}^{-1}\, (\mathbf{y} - \mathbf{b}), \mathbf{A} - \mathbf{C}\, \mathbf{B}^{-1} \mathbf{C}^T) \\ \mathbf{y} \,|\, \mathbf{x} &\sim \mathrm{N}(\mathbf{b} + \mathbf{C}^T\, \mathbf{A}^{-1}\, (\mathbf{x} - \mathbf{a}), \mathbf{B} - \mathbf{C}^T\, \mathbf{A}^{-1}\, \mathbf{C}). \end{aligned} \tag{A.5}$$

References

Akashi, H. and Kumamoto, H. (1977). Random sampling approach to state estimation in switching environments. *Automatica*, 13:429–434.

Anderson, B. D. O. and Moore, J. B. (1979). *Optimal Filtering*. Prentice-Hall.

Anderson, R. M. and May, R. M. (1991). *Infectious Diseases of Humans: Dynamics and Control*. Oxford University Press, Oxford.

Andrieu, C., de Freitas, N., and Doucet, A. (2002). Rao-Blackwellised particle filtering via data augmentation. In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14*. MIT Press.

Arasaratnam, I. and Haykin, S. (2009). Cubature Kalman filters. *IEEE Transactions on Automatic Control*, 54(6):1254–1269.

Bar-Shalom, Y. and Li, X.-R. (1995). *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS.

Bar-Shalom, Y., Li, X.-R., and Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation*. Wiley, New York.

Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. Springer.

Bernardo, J. M. and Smith, A. F. M. (1994). *Bayesian Theory*. John Wiley & Sons.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Blackman, S. and Popoli, R. (1999). *Design and Analysis of Modern Tracking Systems*. Artech House Radar Library.

Bucy, R. S. and Joseph, P. D. (1968). *Filtering for Stochastic Processes with Applications to Guidance*. John Wiley & Sons.

Candy, J. V. (2009). *Bayesian Signal Processing - Classical, Modern, and Particle Filtering Methods*. Wiley.

Casella, G. and Robert, C. P. (1996). Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94.

Challa, S., Morelande, M. R., Mušicki, D., and Evans, R. J. (2011). *Fundamentals of Object Tracking*. Cambridge University Press.

Chen, R. and Liu, J. S. (2000). Mixture Kalman filters. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 62(3):493–508.

Cox, H. (1964). On the estimation of state variables and parameters for noisy dynamic systems. *IEEE Transactions on Automatic Control*, 9(1):5–12.

Crassidis, J. L. and Junkins, J. L. (2004). *Optimal Estimation of Dynamic Systems*. Chapman & Hall/CRC.

Crisan, D. and Rozovskii, B., editors (2011). *The Oxford Handbook of Nonlinear Filtering*. Oxford.

Deisenroth, M. P., Huber, M. F., and Hanebeck, U. D. (2009). Analytic moment-based Gaussian process filtering. In *Proceedings of the 26th International Conference on Machine Learning*.

Doucet, A., de Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*. Springer, New York.

Doucet, A., Godsill, S. J., and Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208.

Fearnhead, P. and Clifford, P. (2003). On-line inference for hidden Markov models via particle filters. *J. R. Statist. Soc. B*, 65(4):887–899.

Fong, W., Godsill, S. J., Doucet, A., and West, M. (2002). Monte Carlo smoothing with application to audio signal enhancement. *IEEE Transactions on Signal Processing*, 50(2):438–449.

Gelb, A. (1974). *Applied Optimal Estimation*. The MIT Press, Cambridge, MA.

Gelb, A. and Vander Velde, W. (1968). *Multiple-Input Describing Functions and Nonlinear System Design*. McGraw Hill.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. R. (1995). *Bayesian Data Analysis*. Chapman & Hall.

Gilks, W., Richardson, S., and Spiegelhalter, D., editors (1996). *Markov Chain Monte Carlo in Practice*. Chapman & Hall.

Godsill, S. J., Doucet, A., and West, M. (2004). Monte Carlo smoothing for nonlinear time series. *Journal of the American Statistical Association*, 99(465):156–168.

Godsill, S. J. and Rayner, P. J. (1998). *Digital Audio Restoration: A Statistical Model Based Approach*. Springer-Verlag.

Golub, G. H. and Welsch, J. H. (1969). Calculation of Gauss quadrature rules. *Mathematics of Computation*, 23(106):221–230.

Gonzalez, R. C. and Woods, R. E. (2008). *Digital Image Processing*. Prentice Hall, 3rd edition.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEEE Proceedings on Radar and Signal Processing*, volume 140, pages 107–113.

Grewal, M., Miyasako, R., and Smith, J. (1988). Application of fixed point smoothing to the calibration, alignment and navigation data of inertial navigation systems. In *Position Location and Navigation Symposium*, pages 476–479.

Grewal, M. S. and Andrews, A. P. (2001). *Kalman Filtering, Theory and Practice Using MATLAB*. Wiley, New York.

Grewal, M. S., Weill, L. R., and Andrews, A. P. (2001). *Global Positioning Systems, Inertial Navigation and Integration*. Wiley, New York.

Hartikainen, J. and Särkkä, S. (2010). Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *Proceedings of IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 379–384.

Hauk, O. (2004). Keep it simple: a case for using classical minimum norm estimation in the analysis of EEG and MEG data. *NeuroImage*, 21(4):1612–1621.

Hayes, M. H. (1996). *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, Inc.

Hiltunen, P., Särkkä, S., Nissila, I., Lajunen, A., and Lampinen, J. (2011). State space regularization in the nonstationary inverse problem for diffuse optical tomography. *Inverse Problems*, 27:025009.

Ho, Y. C. and Lee, R. C. K. (1964). A Bayesian approach to problems in stochastic estimation and control. *IEEE Transactions on Automatic Control*, 9:333–339.

Hürzeler, M. and Kunsch, H. R. (1998). Monte Carlo approximations for general state-space models. *Journal of Computational and Graphical Statistics*, 7(2):175–193.

Ito, K. and Xiong, K. (2000). Gaussian filters for nonlinear filtering problems. *IEEE*

Transactions on Automatic Control, 45(5):910–927.

Jazwinski, A. H. (1966). Filtering for nonlinear dynamical systems. *IEEE Transactions on Automatic Control*, 11(4):765–766.

Jazwinski, A. H. (1970). *Stochastic Processes and Filtering Theory*. Academic Press, New York.

Julier, S. J. and Uhlmann, J. K. (1995). A general method of approximating nonlinear transformations of probability distributions. Technical report, Robotics Research Group, Department of Engineering Science, University of Oxford.

Julier, S. J. and Uhlmann, J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422.

Julier, S. J., Uhlmann, J. K., and Durrant-Whyte, H. F. (1995). A new approach for filtering nonlinear systems. In *Proceedings of the 1995 American Control, Conference, Seattle, Washington*, pages 1628–1632.

Julier, S. J., Uhlmann, J. K., and Durrant-Whyte, H. F. (2000). A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482.

Kaipio, J. and Somersalo, E. (2005). *Statistical and Computational Inverse Problems*. Number 160 in Applied mathematical Sciences. Springer.

Kalman, R. E. (1960a). Contributions to the theory of optimal control. *Boletin de la Sociedad Matematica Mexicana*, 5(1):102–119.

Kalman, R. E. (1960b). A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82:35–45.

Kalman, R. E. and Bucy, R. S. (1961). New results in linear filtering and prediction theory. *Transactions of the ASME, Journal of Basic Engineering*, 83:95–108.

Kaplan, E. D. (1996). *Understanding GPS, Principles and Applications*. Artech House, Boston, London.

Kelly, C. N. and Anderson, B. D. O. (1971). On the stability of fixed-lag smoothing algorithms. *Journal of Franklin Institute*, 291(4):271–281.

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5:1–25.

Kotecha, J. H. and Djuric, P. M. (2003). Gaussian particle filtering. *IEEE Transactions on Signal Processing*, 51(10).

Kuznetsov, P. I., Stratonovich, R. L., and Tikhonov, V. I. (1960). Quasi-moment functions in the theory of random processes. *Theory of Probability and its Applications*, V(1):80–97.

Lee, R. C. K. (1964). *Optimal Estimation, Identification and Control*. M.I.T. Press.

Lefebvre, T., Bruyninckx, H., and Schutter, J. D. (2002). Comment on "a new method for the nonlinear transformation of means and covariances in filters and estimators" [and authors' reply]. *IEEE Transactions on Automatic Control*, 47(8):1406–1409.

Lewis, F. L. (1986). *Optimal Estimation with an Introduction to Stochastic Control Theory*. John Wiley & Sons.

Lin, F.-H., Wald, L. L., Ahlfors, S. P., Hämäläinen, M. S., Kwong, K. K., and Belliveau, J. W. (2006). Dynamic magnetic resonance inverse imaging of human brain function. *Magnetic Resonance in Medicine*, 56:787–802.

Lindsten, F. and Schön, T. B. (2011). Rao-Blackwellized particle smoothers for mixed linear/nonlinear state-space models. Technical Report LiTH-ISY-R-2018, Linköpings Universitet.

Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. Springer.

Liu, J. S. and Chen, R. (1995). Blind deconvolution via sequential imputations. *Journal of* *the American Statistical Association*, 90(430):567–576.

MacKay, D. J. C. (1998). Introduction to Gaussian processes. In Bishop, C. M., editor, *Neural Networks and Machine Learning*, pages 133–165. Springer-Verlag.

Maybeck, P. (1979). *Stochastic Models, Estimation and Control, Volume 1*. Academic Press.

Maybeck, P. (1982a). *Stochastic Models, Estimation and Control, Volume 2*. Academic Press.

Maybeck, P. (1982b). *Stochastic Models, Estimation and Control, Volume 3*. Academic Press.

Meditch, J. (1969). *Stochastic Optimal Linear Estimation and Control*. McGraw-Hill.

Milton, J. S. and Arnold, J. C. (1995). *Introduction to Probability and Statistics, Principles and Applications for Engineering and the Computing Sciences*. McGraw-Hill, Inc.

Moore, J. and Tam, P. (1973). Fixed-lag smoothing of nonlinear systems with discrete measurement. *Information Sciences*, 6:151–160.

Moore, J. B. (1973). Discrete-time fixed-lag smoothing algorithms. *Automatica*, 9(2):163–174.

Murray, J. D. (1993). *Mathematical Biology*. Springer, New York.

Nørgaard, M., Poulsen, N. K., and Ravn, O. (2000). New developments in state estimation for nonlinear systems. *Automatica*, 36(11):1627 – 1638.

O'Hagan, A. (1991). Bayes-Hermite quadrature. *Journal of Statistical Planning and Inference*, 29:245–260.

Pikkarainen, H. (2005). *A Mathematical Model for Electrical Impedance Process Tomography*. Doctoral dissertation, Helsinki University of Technology.

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599.

Proakis, J. G. (2001). *Digital Communications*. McGraw-Hill, 4th edition.

Punskaya, E., Doucet, A., and Fitzgerald, W. J. (2002). On the use and misuse of particle filtering in digital communications. In *EUSIPCO*.

Raiffa, H. and Schlaifer, R. (2000). *Applied Statistical Decision Theory*. John Wiley & Sons, Wiley Classics Library.

Rauch, H. E. (1963). Solutions to the linear smoothing problem. *IEEE Transactions on Automatic Control*, AC-8:371–372.

Rauch, H. E., Tung, F., and Striebel, C. T. (1965). Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450.

Ristic, B., Arulampalam, S., and Gordon, N. (2004). *Beyond the Kalman Filter*. Artech House, Norwood, MA.

Sage, A. P. and Melsa, J. L. (1971). *Estimation Theory with Applications to Communications and Control*. McGraw-Hill Book Company.

Särkkä, S. (2008). Unscented Rauch-Tung-Striebel smoother. *IEEE Transactions on Automatic Control*, 53(3):845–849.

Särkkä, S. (2011). Linear operators and stochastic partial differential equations in Gaussian process regression. In *Proceedings of ICANN*.

Särkkä, S., Bunch, P., and Godsill, S. J. (2012a). A backward-simulation based Rao-Blackwellized particle smoother for conditionally linear Gaussian models. In *Proceedings of SYSID 2012*.

Särkkä, S. and Hartikainen, J. (2010a). On Gaussian optimal smoothing of non-linear state space models. *IEEE Transactions on Automatic Control*, 55(8):1938–1941.

Särkkä, S. and Hartikainen, J. (2010b). Sigma point methods in optimal smoothing of non-linear stochastic state space models. In *Proceedings of IEEE International Workshop on*

Machine Learning for Signal Processing (MLSP), pages 184–189.

Särkkä, S. and Hartikainen, J. (2012). Infinite-dimensional Kalman filtering approach to spatio-temporal Gaussian process regression. In *Proceedings of AISTATS 2012*.

Särkkä, S., Solin, A., Nummenmaa, A., Vehtari, A., Auranen, T., Vanni, S., and Lin, F.-H. (2012b). Dynamic retrospective filtering of physiological noise in BOLD fMRI: DRIFTER. *NeuroImage*, 60:1517–1527.

Särkkä, S., Vehtari, A., and Lampinen, J. (2007a). CATS benchmark time series prediction by Kalman smoother with cross-validated noise density. *Neurocomputing*, 70(13–15):2331–2341.

Särkkä, S., Vehtari, A., and Lampinen, J. (2007b). Rao-Blackwellized particle filter for multiple target tracking. *Information Fusion Journal*, 8(1):2–15.

Sarmavuori, J. and Särkkä, S. (2012). Fourier-Hermite Kalman filter. *IEEE Transactions on Automatic Control*. (in press).

Schön, T., Gustafsson, F., and Nordlund, P.-J. (2005). Marginalized particle filters for mixed linear/nonlinear state-space models. *IEEE Transactions on Signal Processing*, 53(7):2279–2289.

Shiryaev, A. N. (1996). *Probability*. Springer.

Stengel, R. F. (1994). *Optimal Control and Estimation*. Dover Publications, New York.

Stone, L. D., Barlow, C. A., and Corwin, T. L. (1999). *Bayesian Multiple Target Tracking*. Artech House, Boston, London.

Storvik, G. (2002). Particle filters in state space models with the presence of unknown static parameters. *IEEE Transactions on Signal Processing*, 50(2):281–289.

Stratonovich, R. L. (1968). *Conditional Markov Processes and Their Application to the Theory of Optimal Control*. American Elsevier Publishing Company, Inc.

Tam, P., Tam, D., and Moore, J. (1973). Fixed-lag demodulation of discrete noisy measurements of FM signals. *Automatica*, 9:725–729.

Tarantola, A. (2004). *Inverse Problem Theory and Methods for Model Parameter Estimation*. SIAM.

Titterton, D. H. and Weston, J. L. (1997). *Strapdown Inertial Navigation Technology*. Peter Pregrinus Ltd.

Van der Merwe, R., Freitas, N. D., Doucet, A., and Wan, E. (2001). The unscented particle filter. In *Advances in Neural Information Processing Systems 13*.

Van der Merwe, R. and Wan, E. (2003). Sigma-point Kalman filters for probabilistic inference in dynamic state-space models. In *Proceedings of the Workshop on Advances in Machine Learning*.

Van Trees, H. L. (1968). *Detection, Estimation, and Modulation Theory Part I*. John Wiley & Sons, New York.

Van Trees, H. L. (1971). *Detection, Estimation, and Modulation Theory Part II*. John Wiley & Sons, New York.

Vauhkonen, M. (1997). *Electrical impedance tomography and prior information*. PhD thesis, Kuopio University.

Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, II-13(2).

Wan, E. A. and Van der Merwe, R. (2001). The unscented Kalman filter. In Haykin, S., editor, *Kalman Filtering and Neural Networks*, chapter 7. Wiley.

West, M. and Harrison, J. (1997). *Bayesian Forecasting and Dynamic Models*. Springer-Verlag.

Wiener, N. (1950). *Extrapolation, Interpolation and Smoothing of Stationary Time Series with Engineering Applications*. John Wiley & Sons, Inc., New York.

Wu, Y., Hu, D., Wu, M., and Hu, X. (2005). Unscented Kalman filtering for additive noise case: Augmented versus nonaugmented. *IEEE Signal Processing Letters*, 12(5):357–360.

Wu, Y., Hu, D., Wu, M., and Hu, X. (2006). A numerical-integration perspective on Gaussian filters. *IEEE Transactions on Signal Processing*, 54(8):2910–2921.

Index