State Estimation Filters

Written Preliminary Exam II Report

Radoslav Ivanov

Department of Computer and Infomation Science
University of Pennsylvania

Abstract

Precise state estimation is crucial for any autonomous system as it enables the system to provide appropriate control inputs to its actuators. However, estimation is challenging as it must be able to handle imprecise system models (often described by complex non-linear functions) as well as noisy sensor measurements in order to compute the system's state. In this work, we explore three approaches to state estimation filtering that have different theoretical guarantees and estimation accuracies depending on the system in question (e.g., a linear system with Gaussian noise). We describe the theoretical foundations of each filter and track its evolution over time. Finally, the differences and similarities between the filters are emphasized, and some appropriate applications are discussed.

1 Introduction

State estimation is one of the fundamental problems in the study of autonomous control systems. Precise state estimation is a necessary condition for effective control; it allows a system to have accurate knowledge of its current state and, consequently, to make informed decisions when choosing future control inputs.

At a high level, the state estimation filtering problem is as follows. The system in question (e.g., an autonomous vehicle) would like to achieve a certain goal (e.g., reach a destination); to do this, the system needs to know its state (e.g., position, velocity) in order to apply the necessary control inputs. To compute its state, the system has access to measurements (e.g., GPS and odometry measurements) of its state; since these measurements can often be faulty, the system's goal is to use all of its available information about its sensor models and about its own dynamics in order to obtain a precise estimate of its state.

In this work, we explore different approaches to state estimation that have been developed in the literature. We note that the power and complexity of these

approaches vary greatly depeding on the amount of knowledge about the system. In particular, if nothing is known about the evolution of the system over time and the only available information is the relation between the measurements and the state, then hardly any improvements upon the measurements can be made. If, on the other hand, more precise system and sensor models are available, then it is possible to greatly improve the state estimate and to tighten the set of possible values that the state can take.

Therefore, for any system under consideration, it is necessary that dynamical and observation models be developed in order for the system to be accurately analyzed. These models are necessarily abstractions of reality; developing perfect models is rarely possible and is also impractical – modeling events that have a minor effect on the system's operation may lead to complex functional models with no major improvement in accuracy; thus, such events are usually included as noise that may perturb the system's nominal behavior.

Multiple approaches to state estimation filtering have been explored depending on the functional form of the dynamical and observation model as well as on the assumptions about the noise distribution (e.g., probabilistic, bounded). We describe three classes of algorithms that cover most models and applications observed in practice today.

The first class of algorithms are Kalman filters. The classical Kalman filter [21] is the best linear unbiased estimator (BLUE) for linear systems with Gaussian process and measurement noise. The Kalman filter's closed-form recursive equations have turned it into arguably the most popular and widely used estimator, with applications ranging from the aerospace and aircraft industries to seismology and weather forecasting [13]. In addition to the original algorithm, we briefly mention some popular extensions (i.e., the extended Kalman filter (EKF) [17] and a Gaussian mixture filter [1]), with an emphasis on a distributed version of the filter [31].

An alternative estimation algorithm that can be used for non-linear systems with non-Gaussian noise is the Markov Chain Monte Carlo filter [12], also known as the particle filter. Particle filters work by simulating the system evolution multiple times and choosing the state estimate as a weighted average of all simulations (particles). They have been shown to work well in multiple robotic applications, notably the simultaneous localization and mapping (SLAM) problem [28]. In addition, it is possible to combine Kalman and particle filters in order to leverage the advantages of both approaches in a Rao-Blackwellised particle filter [10].

Finally, we investigate another approach to perform state estimation in non-linear systems, namely set membership filtering [26]. In this line of works, the system is assumed to have bounded process and measurement noise. Thus it is possible to construct a set containing all possible values of the state at the current time. Set membership filters have been shown to outperform other non-linear estimators (e.g., the EKF) in highly non-linear systems with bounded noise [27].

This paper is organized as follows. In Section 2 we formalize the state estimation filtering problem, while Section 3 discusses porssible solutions at

a high level. The following three sections, Section 4, Section 5, and Section 6, describe each of the three classes of filters explored in this paper, namely Kalman filters, particle filters and set membership filters, respectively. Finally, Section 7 concludes the work.

2 State Estimation Filtering Problem

The state estimation filtering problem is as follows. Consider a system with known dynamics of the form

$$x_{k+1} = f(x_k) + w_k, \tag{1}$$

where $x_k \in \mathbb{R}^n$ is the system state at time k (e.g., position, velocity), $w_k \in \mathbb{R}^n$ is process noise (as argued above, w_k contains unmodeled dynamics, e.g., friction for automotive systems) and f is a (possibly nonlinear) function describing the evolution of the state.[1]

At each point in time k, the system has access to measurements of the form

$$y_k = g(x_k) + v_k, \tag{2}$$

where $y_k \in \mathbb{R}^p$ are the available measurements, $v_k \in \mathbb{R}^p$ is measurement noise (due to sensor imprecisions, e.g., GPS measurements jumping greatly in an urban environment), and g is a (possibly nonlinear) function mapping the states to the available measurements.

Problem. *Consider the system defined in* (1)-(2)*. Given measurements* $y_1, \ldots, y_k$*, the state estimation filtering problem is to compute an optimal estimate* $\hat{x}_k$ *of the state* x_k *at time* k*.*

Note that formalizing a notion of optimality depends on the actual system model that is used. For example, if x_k is a random variable, then a possible definition of optimality is that "$\hat{x}_k$ is an unbiased estimator of x_k with minimal variance among all other unbiased estimators of x_k." On the other hand, if x_k is driven by bounded noise, one could develop a set of possible values for x_k; in this case, an optimality criterion may be that this set is the smallest set that is guaranteed to contain the true state.

Note also that in addition to the filtering problem, i.e., the estimation of x_k, one may also be interested in the smoothing problem, i.e., the estimation of all states from x_1 through x_k (e.g., in SLAM when the map is part of the state, and it is not changing over time) as well as the prediction problem, i.e., the estimation of x_{k+1} and later states (e.g., as in model-predictive control). While related, these problems require different techniques, hence smoothing and prediction are not discussed in this paper.

[1] Note that in most systems x_{k+1} also depends on the input at time k, u_k. To keep notation simple, u_k is not considered in this work; however, all described techniques can be straightforwardly enhanced to incorporate inputs as well.

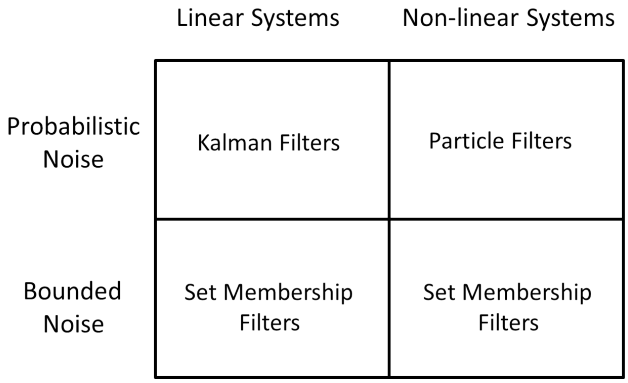


Figure 1: An overview of the state estimation filtering problem state space and some of the respective filters used in each scenario.

3 General Approach to State Estimation

As noted above, the first consideration when performing state estimation is the system model's functional form. There are two aspects of the functional form that mainly affect the analysis of such systems – 1) is the system model linear vs. non-linear in the state, and 2) is the noise modeled as a random variable vs. a non-random bounded variable. Further refinements can be made as well (e.g., quadratic systems, polynomial systems) but this is the most general classification of the systems considered in existing literature.

In this section, we explore the state space determined by the above classification at a high level; for easy reference, it is also summarized in Figure 1. Note that in this section we split the models according to their noise assumptions only; the general concepts discussed below are not affected by the linearity of the system in question. The following sections will focus on specific algorithms addressing each aspect of the state space in Figure 1.

3.1 Probabilistic Noise

By far the most popular class of models in the literature are those that assume that noise is a random variable with a known probability distribution. As argued in previous works [21], in many practical applications the process noise tends to have an empirical distribution that is close to Gaussian (e.g., noise voltage in resistors due to thermal agitation). Multiple other probability distributions have been considered as well depending on the application (e.g., truncated Gaussian distributions [35] for turbofan engine health estimation).

3.1.1 Optimality and Cost Functions

In the probabilistic setting, there are multiple ways to define an optimal estimate. One desired property of an estimator is that it be unbiased, i.e., the estimate eventually converges to the true state. However, this is a limit property (i.e., holds as time goes to infinity, similar to the spirit of the law of large numbers) and it does not provide any guarantees about estimates at any specific points in time. Thus, the most common requirement of good estimators is that they minimize some distance between the estimate and the true state at any point in time.

An example distance function is the absolute value of the difference $e = \hat{x}_k - x_k$ between the estimate and the true state, i.e., $c(e) = |e|$, where for simplicity x_k is assumed to be one-dimensional, though the function can be straightforwardly extended to arbitrary dimensions. Thus, if an estimator minimizes c for all k, then the estimate is always equal to the true state. The function c is just one example of a cost function; more generally, a cost function can be defined as follows:

Definition (Cost Function)**.** *A function f_c is said to be a **cost function** if it is:*

1. *Symmetric around 0, i.e., $f_c(e) = f_c(-e)$.*
2. *Nondecreasing for positive numbers, i.e., $0 < e_1 \leq e_2 \implies f_c(e_1) \leq f_c(e_2)$.*

However, a problem arises when directly manipulating a cost function due to the fact the true state x_k is never known, hence the function cannot be minimized for the true state. Thus, a logical replacement is the expected value of $f_c(e)$, i.e., $\mathbb{E}[f_c(e)]$, which is well defined because x_k is a random variable. Thus, we can now define an optimal estimator as follows.

Definition (Optimal Estimator)**.** *An estimator of x_k is **optimal** with respect to a cost function f_c if its estimate $\hat{x}_k$ minimizes the expected cost $\mathbb{E}[f_c(\hat{x}_k - x_k)]$ for all k.*

Researchers have considered different choices for f_c but the one that has attracted the most attention, mostly due to cleaner mathematical derivations, is the quadratic cost function $f_c(e) = e^2$. This choice has been used in one of the classical filters, namely the Wiener filter [39], to derive an optimal estimator for linear system models (i.e., f and g in (1)-(2) are linear functions) and arbitrary noise distributions. Somewhat interestingly, the same optimal estimator results when the noise distribution is Gaussian and the cost function is arbitrary.

In order to maintain consistency with the majority of works in the area, in this paper we also assume that the cost function is $f_c(e) = e^2$ and/or the noise distribution is Gaussian, both of which lead to the following optimal estimator.

Theorem 1 (Gaussian/Min-Squared-Error Optimal Estimator [21])**.** *If $f_c(e) = e^2$ or if the process and sensor noises in* (1)-(2) *have Gaussian distributions, the optimal estimate of x_k at each k is $\mathbb{E}[x_k \mid y_1, \ldots, y_k]$, i.e., the conditional expectation of x_k given all available measurements.*

3.1.2 Bayes Filters

Having fixed the optimal estimator as in Theorem 1, it remains to show how to compute this filter for different systems. Many covered by Theorem 1 can be classified as what is referred to as a Bayes filter, i.e., one that uses a prior estimate and recursively computes the current estimate through a predict and update stage. Intuitively, the predict stage computes the estimate of the state in between measurements and the update stage computes the state estimate when a new measurement arrives. Both classes of probabilistic filters considered in this paper, i.e., Kalman and particle filters, are Bayes filters.

Bayes filters are appealing because they are recursive, which makes them easy to understand and, more importantly, to implement in real time. In other words, each predict stage uses the estimate of the previous update stage and each update stage uses the estimate of the previous predict stage. To formalize the filter, we first define the predict and update stage estimates before providing the actual filter equations.

Note that conditional densities are used in all derivations instead of conditional expectations because densities are somewhat easier to analyze. For any random variable X, its probability density function (pdf) is a function p such that

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} xp(x)dx,$$

i.e., the expected value of X can be directly obtained from its pdf. The density of $\mathbb{E}[x_k \mid y_1, \ldots, y_k]$ (which is a random variable) is referred to as a conditional density.

Definition (Predicted conditional density)**.** *The predicted conditional density of the Bayes filter is defined as the conditional density of x_k given measurements $y_1, \ldots, y_{k-1}$:*

$$p_{k|k-1}(x_k) := p(x_k \mid y_1, \ldots, y_{k-1}). \tag{3}$$

Definition (Updated conditional density)**.** *The updated conditional density of the Bayes filter (also referred to as a **posterior density**) is defined as the conditional density of x_k given measurements $y_1, \ldots, y_k$:*

$$p_{k|k}(x_k) := p(x_k \mid y_1, \ldots, y_k). \tag{4}$$

Note that the reason Bayes filters are recursive and easy to implement is that the systems in consideration satisfy the Markov property. Intuitively, the Markov property says that the current state x_k is only a function of the previous state x_{k-1}, regardless of the values of the states and measurements before that. A similar statement exists for the current measurements.

Definition (Markov Property)**.** *In a system satisfying the Markov property, the current state x_k, when conditioned on x_{k-1}, is independent of x_p for all*

$p < k-1$ *and of* y_q *for all* $q \leq k$*:*

$$p(x_k \mid x_{k-1}, x_1, \ldots, x_{k-2}, y_1, \ldots, y_k) = p(x_k \mid x_{k-1}). \tag{5}$$

Similarly, the current measurement y_k *is independent of previous states and measurements when conditioned on the current state* x_k*:*

$$p(y_k \mid x_k, x_1, \ldots, x_{k-1}, y_1, \ldots, y_{k-1}) = p(y_k \mid x_k). \tag{6}$$

We are now ready to describe the equations of the general Bayes filter. Note that the following derivation is borrowed and adapted from [37].

Theorem 2 (Bayes filter)**.** *Given a prior density* $p_{k-1|k-1}$*, the predicted and updated densities, respectively, are computed as follows:*

$$p_{k|k-1}(x_k) = \int p_f(x_k \mid x_{k-1}) p_{k-1|k-1}(x_{k-1}) dx_{k-1}$$
$$p_{k|k}(x_k) = \eta_k p_o(y_k \mid x_k) p_{k|k-1}(x_k),$$

where p_f *is the conditional density of the state* x_k *given* x_{k-1} *(as derived from* (1)*),* p_o *is the conditional density of the measurement* y_k *given* x_k *(as derived from* (2)*) and* $\eta_k = \int p(y_k \mid z) p_{k|k-1}(z) dz$ *is a normalization constant that ensures the posterior integrates to 1.*

Remark. *This algorithm is initialized by assuming a prior distribution on* x_0*.*

Proof. The predicted density is derived as follows:

$$\begin{aligned} p_{k|k-1}(x_k) &= p(x_k \mid y_1, \ldots, y_{k-1}) \\ &= \int p(x_k, x_{k-1} \mid y_1, \ldots, y_{k-1}) dx_{k-1} \\ &= \int p(x_k \mid x_{k-1}, y_1, \ldots, y_{k-1}) p(x_{k-1} \mid y_1, \ldots, y_{k-1}) dx_{k-1} \\ &= \int p_f(x_k \mid x_{k-1}) p_{k-1|k-1}(x_{k-1}) dx_{k-1}. \end{aligned}$$

The second equality is due to the "marginalization" property of the joint distribution, the third follows from Bayes rule, and the fourth is derived from the Markov property and the definition of the prior density.

The updated density can be computed as follows:

$$\begin{aligned} p_{k|k}(x_k) &= p(x_k \mid y_1, \ldots, y_k) \\ &= \frac{p(y_k \mid x_k, y_1, \ldots, y_{k-1}) p(x_k \mid y_1, \ldots, y_{k-1})}{p(y_k \mid y_1, \ldots, y_{k-1})} \\ &= \frac{p_o(y_k \mid x_k) p_{k|k-1}(x_k)}{\int p_o(y_k \mid z) p_{k|k-1}(z) dz}. \end{aligned}$$

The second equality is Bayes rule, and the third is the Markov property, the definition of the predicted density and the marginalization property in the denominator. □

As can be observed in the above result, each of the two equations is a function of the previous stage of the filter, thus illustrating its recursive nature. The derived integrals do not have closed-form solutions for arbitrary systems, yet an important exception is the linear Gaussian case which leads to the Kalman filter described in Section 4. Monte-Carlo approximations of these integrals can also be computed, as is done in the particle filter, which is the topic of Section 5.

3.2 Bounded Noise

Although the probabilistic noise assumption may be reasonable in many scenarios, there exist cases where it does not hold or where the noise distribution has a complicated functional form that is difficult to analyze. In such cases, researchers attempt to determine a global bound on the noise and try to estimate the state based on this bound. These approaches are referred to as set membership methods because the bounds result in sets of possible values for the state.

3.2.1 Optimality

Defining optimality in this setting is also not straightforward as it is not clear how to compare two state estimates when the true value is unknown. A popular approach is to say an estimator is optimal if it minimizes the worst-case error (recall the error is the difference $e = \hat{x}_k - x_k$) as compared with some class of estimators. This class of estimators is obtained by making assumptions about the system dynamics – if no such assumptions are made, then the worst-case error can be arbitrarily large since multiple functions could map the available measurements to the true state. Thus, in set membership filtering, the class of possible dynamics (hence, the class of possible estimators) is restricted to a broad, but analytically manageable, set of functions (e.g., continuously differentiable functions of the measurements [26]).

With this notion in mind, we now formalize optimality. Let $\mathcal{K}$ be the set of all possible system models. If we assume that the measurement noise is bounded by δ (i.e., $\|v_k\| \leq \delta$), then the worst-case error $e_{wc}(f, y_1, \ldots, y_k)$ of an estimator f, given measurements $y_1, \ldots, y_k$, is defined as:

$$e_{wc}(f, y_1, \ldots, y_k) = \sup_{\hat{f} \in \mathcal{K}} \sup_{\tilde{y}_i \in \mathcal{B}_\delta(y_i)} \|f(y_1, \ldots, y_k) - \hat{f}(\tilde{y}_1, \ldots, \tilde{y}_k)\|, \qquad (7)$$

where $\mathcal{B}_\delta(y_i)$ is a ball of radius δ centered at y_i, $\tilde{y}_i$ represent all possible values of the true state given the received measurements, and $\hat{f}$ represent all possible system dynamics. Thus, e_{wc} is indeed a worst-case measure – it is achieved when both the actual states are far from the measurements and the true dynamics are very different from f.

Definition (Optimal Estimator (Bounded Noise))**.** *An estimator f is called **optimal** with respect to a class of estimators $\mathcal{K}$ if it minimizes the worst-case error e_{wc} as defined in* (7)*.*

Note that this is an intrinsically different definition from the probabilistic one, where only the expected error is minimized and no guarantees are made about the worst case. On other hand, estimators satisfying this definition always have bounded performance but may achieve worse average performance overall.

A popular approach to set membership filtering is discussed in Section 6. We discuss a possible definition for the set $\mathcal{K}$ and show how to find an optimal estimator in $\mathcal{K}$.

4 Kalman Filters

This section describes the original Kalman filter algorithm and then presents several important extensions, notably the extended Kalman filter and the Kalman consensus filter, a distributed version of the algorithm.

4.1 Classical Kalman Filter

As developed in 1960 by Kalman [21], the Kalman filter is a modified version of the Wiener filter [39]. Whereas the Wiener filter computes the estimate using transfer functions, the Kalman filter introduces the notion of state and state transition. The Kalman filter is developed for linear systems, i.e., (1)-(2) now become

$$\begin{aligned} x_{k+1} &= A_k x_k + w_k \\ y_k &= C_k x_k + v_k, \end{aligned} \tag{8}$$

where A_k and C_k are matrices of appropriate dimensions, possibly changing over time. The Kalman filter is a mean-squared-error filter, hence it satisfies the conditions in Theorem 1 and thus computes the conditional expectation of the state:

$$\mathbb{E}[x_k \mid y_1, \ldots, y_k].$$

Since the conditional expectation is also the optimal solution in the case when w_k and v_k have Gaussian distributions, a Gaussian noise assumption is usually implied whenever one refers to the Kalman filter (even though this is a just a special case of the filter). As this has become the norm in the related literature, in this work we also assume that $w_k \sim \mathcal{N}(0, Q)$, $v_k \sim \mathcal{N}(0, R)$, and that the initial condition $x_0 \sim \mathcal{N}(\mu_0, \Sigma_0)$.

Given the system model in (8), the Kalman filter is a set of recursive equations used to compute the conditional expectation $\mathbb{E}[x_k \mid y_1, \ldots, y_k]$. Due to its popularity, it has undergone many improvements in presentation and derivation in order to improve researchers' understanding. In this paper, we briefly present the initial approach taken by Kalman before providing the Bayes filter formulation that we believe is the most intuitive version of the Kalman filter.

4.1.1 Original Algorithm

In his original paper [21], Kalman notes that $\mathbb{E}[x_k \mid y_1, \ldots, y_k]$ is the orthogonal projection of x_k onto the subspace $\mathcal{Y}_k$ spanned by the vectors $y_1, \ldots, y_k$. The idea of his approach is that $\mathcal{Y}_k$ can be represented as the combination of $\mathcal{Y}_{k-1}$ and $\mathcal{Z}_k$, where $\mathcal{Z}_k$ is the subspace spanned by y_k. This separation allows one to exploit the subspace structure recursively and thus design an efficient filtering algorithm. While still recursive, this derivation does not have update and predict stages, hence it is not a commonly used one. We do not include it in this paper in the interest of space.

4.1.2 Bayes Formulation of the Kalman Filter

The Bayes filter derivation of the Kalman filter is a direct application of Theorem 2. The main idea of the derivation is that all conditional densities in Theorem 2 (i.e., $p_f(x_{k+1} \mid x_k), p_o(y_k \mid x_k), p_{k+1|k+1}(x_{k+1}), p_{k+1|k}(x_{k+1})$) are Gaussian pdf's. This means that the Kalman filter always maintains a Gaussian distribution with parameters that can be computed in closed form.

Theorem 3. *Given a linear system as in* (8) *with Gaussian process and measurement noise, and given that the prior on the state $p_{k|k}$ is a Gaussian distribution with mean μ and covariance Σ, the predicted and update densities are computed as follows:*

$$\begin{aligned} p_{k+1|k}(x_{k+1}) &= \phi(x_{k+1}; \mu^p, \Sigma^p) \\ p_{k+1|k+1}(x_{k+1}) &= \phi(x_{k+1}; \mu^u, \Sigma^u), \end{aligned}$$

where $\phi(x_{k+1}; \mu, \Sigma)$ denotes the probability density function of a Gaussian distribution with mean μ and covariance Σ, and

$$\begin{aligned} \mu^p &:= A_k \mu \\ \Sigma^p &:= A_k \Sigma A_k^T + Q \\ \mu^u &:= \mu^p + K(y_{k+1} - C_{k+1}\mu^p) \\ \Sigma^u &:= (I - KC_{k+1})\Sigma^p \\ K &:= \Sigma^p C_{k+1}^T (C_{k+1}\Sigma^p C_{k+1}^T + R)^{-1}. \end{aligned} \tag{9}$$

Remark. *The matrix K is known as the **Kalman gain**. As is apparent from the third equation, it is multiplied by the difference between the received measurements and the expected measurements; this difference is referred to as innovation. Thus, the Kalman gain, which is a function of the state covariance, observation matrix and measurement noise, determines the degree to which innovations affect the estimate.*

Sketch of Proof. The full rigorous proof of the Kalman filter equations is quite lengthy and technical, so only a sketch is provided in the interest of space. A more complete proof can be found in [37]. According to the Bayes filter

equations, the predict step is as follows:

$$\begin{aligned} p_{k+1|k}(x_{k+1}) &= \int p_f(x_{k+1} \mid x_k) p_{k|k}(x_k) dx_k \\ &= \int \phi(x_{k+1}; A_k x_k, Q) \phi(x_k; \mu, \Sigma) dx_k \\ &= \phi(x_{k+1}; A_k \mu, A_k \Sigma A_k^T + Q) \\ &= \phi(x_{k+1}; \mu^p, \Sigma^p), \end{aligned}$$

where the second equality is true because the conditional Gaussian density is linear and the third is true because the integral of the product of two Gaussian densities is again a Gaussian density.

The updated density is derived as follows:

$$\begin{aligned} p_{k+1|k+1}(x_{k+1}) &= \frac{p(y_{k+1} \mid x_{k+1}) p_{k+1|k}(x_{k+1})}{\int p(y_{k+1} \mid z) p_{k+1|k}(z) dz} \\ &= \frac{\phi(y_{k+1}; C_{k+1} x_{k+1}, R) \phi(x_{k+1}; \mu^p, \Sigma^p)}{\int \phi(y_{k+1}; C_{k+1} z, R) \phi(z; \mu^p, \Sigma^p) dz} \\ &= \phi(x_{k+1}; \mu^u, \Sigma^u), \end{aligned}$$

with μ^u and Σ^u defined as above. The second equality is once again due to the linearity of conditional Gaussian expectations. The third is true because the product of two Gaussian densities is again a Gaussian density. □

Thus, the Kalman filter is reduced to just a few matrix equations that can be easily implemented and executed in real time for multidimensional systems. The most computationally (and possibly numerically unstable) calculation is the matrix inverse when computing the Kalman gain – this has led to the development of the Information filter which uses the inverse of the covariance matrix instead of the actual covariance matrix [37]. The Information filter is useful in systems where the covariance matrix contains very small values such that its inverse is close to infinity.

4.1.3 Applications

Linear systems can be found in many branches of engineering; hence the Kalman filter has become arguably the most popular filter and has been applied in numerous fields. The filter's initial intended application was in aerospace [13]. Since then, it has been applied in autonomous navigation (e.g., autopilots in aircraft [36] and road vehicles [23]), seismology [6], radar tracking [24], etc.

4.2 Distributed Kalman Filter

While the centralized Kalman filter has been applied to multiple systems due to its theoretical guarantees, in many modern systems a distributed filter is required. For example, in systems where multiple sensors are spread over a

large area (e.g., in order to monitor methane emissions in landfills [16]) or where multiple robots attempt to collaboratively accomplish a task (e.g., estimate their joint position in a formation [2]), it is often infeasible to transmit all measurements to a centralized location and then compute the estimates in real time. In such cases, each node in the system (i.e., robot, sensor) communicates only with nearby nodes (neighbors) and computes a local estimate of the whole system's state; information from distant nodes is eventually received as well, thus implying that a distributed algorithm might still have good convergence properties.

There exist several distributed versions of the Kalman filter, depending on the application and on the communication model. In this paper, we discuss a popular algorithm called the Kalman Consensus Filter (KCF) [30, 31]. In this version each node in the system sends its state estimate to all of its neighbors; once it receives their estimates, it updates its own estimate of the system's state in order to incorporate the new information. This is a very general model as it does not require great communication or computation capabilities from any node, e.g., it applies to the multi-robot systems described above.

4.2.1 Problem Formulation

Formally, the problem is as follows. Consider a system with linear dynamics as before:

$$x_{k+1} = A_k x_k + w_k. \tag{10}$$

$$\tag{11}$$

The system contains n sensors[2] that measure (parts of) the state with linear observational models:

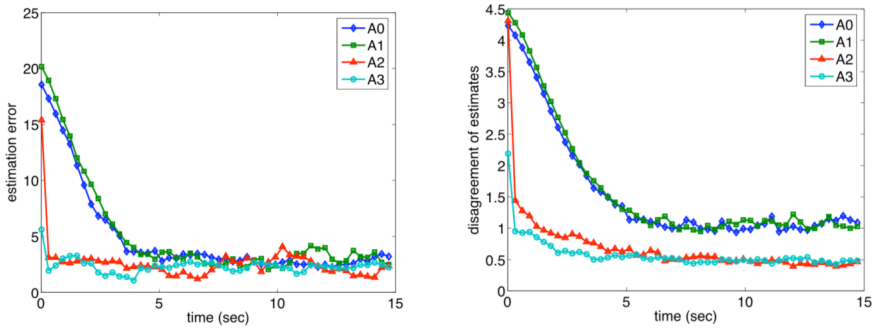
$$y_k^i = C_k^i x_k + v_k^i, \tag{12}$$

where y_k^i is the measurement received by sensor i; similarly C_k^i and v_k^i are the observation matrix and measurement noise, respectively, specific to sensor i. Note that the C_k^i's may not measure the entire state, i.e., the system may not be observable by a single sensor, thus communication is necessary in order for each node to compute a precise state estimate.

It is assumed that each sensor can communicate only with nearby other sensors (the definition of "nearby" depends on the application). The system of sensors is thus formalized as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each sensor i is a vertex $v_i \in \mathcal{V}$, and an edge $(v_i, v_j) \in \mathcal{E}$ exists when v_i and v_j are neighbors, i.e., they can communicate. During each round k, every v_i transmits its predicted estimate $x_{k+1|k}^i$ of the state to all of its neighbors.

The problem addressed by this paper is to develop a filter that is executed at each sensor so that each sensor computes a precise estimate of the entire state.

[2]Note that the word "sensors" can be replaced with "robots" or other devices with sensors that may have their own state, which is also captured in the overall system state.



(a) Average estimation error.

(b) Average disagreement between sensors.

Figure 2: Evaluation of the Kalman Consensus Filter [30].

4.2.2 Approach

The approach took by the author of the KCF is two-fold. First, he suggests that one way of incorporating a received state estimate from another node is through a "consensus gain" matrix, similar to the Kalman gain matrix when a new measurement is received (i.e., matrix K in (9)) [30]. Thus, a consensus update is implemented in addition to the measurement update. For sensor v_i and each neighbor v_j, the consensus update has the form $L(x^i_{k+1|k} - x^j_{k+1|k})$ for some matrix L, i.e., the update equation in the Kalman filter now becomes:

$$\mu^u := \mu^p + K(y_{k+1} - C_{k+1}\mu^p) + L(x^i_{k+1|k} - x^j_{k+1|k}).$$

The author thus investigates the choice of L matrices and derives a form that leads to estimates that converge as time progresses [30]. The actual expression for L and its derivation involve Lyapunov functions and are not included here in the interest of space.

Once the L matrices are fixed, the choice of the Kalman gain matrix K also needs to be considered because the Kalman filter expression for K no longer provides any optimality guarantees. Thus, in the extension of the original KCF [31], the author investigates optimal choices for K. In particular, a value for K is chosen such that it minimizes the mean squared error of the estimate, as discussed in Section 3. This is achieved by finding the value of K that minimizes the trace of the covariance matrix Σ (recall the diagonal elements of Σ are the individual expected squared errors).

4.2.3 Evaluation

The author evaluated the Kalman Consensus Filter in simulation by creating a sensor network with randomly located nodes. The system in consideration consists of two states and evolves according to the following linear model:

$$x_{k+1} = Ax_k + w_k,$$

where A is a given matrix (the full expression for A can be found in [30] but is omitted here to avoid confusion and unnecessary notation). The observation model for each node is:

$$y_k^i = C_k^i x_k + v_k^i,$$

where each C_k^i is either [0 1] or [1 0] so that the system is unobservable by any single node. Yet, by communicating, the nodes can reach a consensus over time and estimate the system's state.

Figure 2 shows the estimation results using the Kalman Consensus Filter; note that several algorithms were developed in the original paper that were not described here, yet the overall trends are the same for all algorithms. In particular, Figure 2a shows that the average estimation error over all nodes decreases as time goes by. In addition, since consensus is arguably equally important in a distributed setting as is the average estimation error, the filter is evaluated by computing the average disagreement between the nodes. As shown in Figure 2b, the average disagreement decreases over time and is close to 0 for the best algorithms.

In conclusion, although this filter does not have the optimality guarantees of the Kalman filter, it is a promising approach to the distributed filtering problem for linear systems.

4.3 Extensions

Although linear Gaussian systems are abundant, most systems in reality are non-linear, and they often do not have Gaussian noise. Wrong assumptions about the system can lead the Kalman filter to diverge and provide estimates with very high errors. Consequently, multiple extensions have been developed to deal with various scenarios encountered in practice. We briefly describe two such extensions in this subsection.

4.3.1 Extended Kalman Filter

The extended Kalman filter (EKF) is an immediate extension of the Kalman filter for non-linear systems. It works by starting with the non-linear model of original system, i.e., (1)-(2), and linearizing the system at each point in time. The idea of the EKF is that if the system is close to linear for small periods of time, then using its linear approximation will not yield large errors.

The EKF works by computing derivatives of the transition and observation functions, i.e., f and g in (1)-(2):

$$A_k = \left.\frac{\partial f(x_k)}{\partial x_k}\right|_{\hat{x}_{k|k}}$$
$$C_k = \left.\frac{\partial g(x_k)}{\partial x_k}\right|_{\hat{x}_{k|k-1}}.$$

A_k and C_k are then used in same way as in the original linear model in (8). The EKF is also a widely used algorithm, with applications in robotics and navigation [37], target tracking [24], etc.

4.3.2 Gaussian Mixture Filter

Another popular extension of the Kalman filter is the Gaussian Mixture (GM) filter. A GM has a probability density function h that is a linear combination of Gaussian densities:

$$h(x) = \sum_{i=1}^{M} w_i \phi(x; \mu_i, \Sigma_i),$$

where w_i are positive weights that sum to 1. GM's have been extensively studied in the literature [17] due to their linear relationship to Gaussians and also due to the fact that, given enough elements in the mixture, they can be used to approximate any distribution with a continuous density.

The GM filter is a straightforward extension of the classical Kalman filter. Due to their linear nature, GM's are propagated with a bank of Kalman filters (one for each element in the mixture), hence the posterior is also a GM, i.e., the filter has a closed-form solution for GM's.

5 Particle Filters

While the Kalman filter and its extensions provide a great set of tools for analyzing linear systems and certain non-linear systems that can be linearly approximated, most systems observed in practice are (highly) non-linear. In addition, the Gaussian noise assumption is often violated as well. Both of these factors cause the integrals in the Bayes filter (Theorem 2) to become intractable.

The rapid development of modern computers in the last few decades has made it possible to approximate the pdf's in Theorem 2 through simulation. Known as Monte Carlo methods, such approaches work by creating multiple simulations (particles) of a system's operation through time, and then analyzing these particles (e.g., computing their moments) as a proxy for the unknown true distribution. The motivation behind this approach is a law-of-large-numbers argument – as the number of particles gets large, their empirical distribution gets close to the true distribution.

5.1 Classical Particle Filter

Similar to the Kalman filter, the particle filter has undergone multiple adaptations and changes throughout its history; its modern development is often credited to Gordon [12]. More recent works have generalized it and have shown its connection to the Bayes filter [10, 37]. The name "particle filter" was first coined by Del Moral [7] and is now widely used in the robotics community [8]. "Particle filter" is also the preferred name in this paper, though the filter is

also referred to in the literature as the Markov Chain Monte Carlo filter or the bootstrap filter [12].

5.1.1 Problem Statement

The particle filter assumes the general non-linear system model with additive noise:

$$\begin{aligned} x_{k+1} &= f(x_k) + w_k \\ y_k &= g(x_k) + v_k, \end{aligned}$$

where w_k and v_k are noises with known distribution (not necessarily Gaussian). The problem addressed by the particle filter is the same recursive filtering problem as in the Kalman filter.

5.1.2 Approach

Intuitively, the particle filter works by simulating the system's operation multiple times, each time drawing a new random number from the process noise's distribution. This way, as the number of particles gets large, the empirical distribution of the particles will approach the true probability distribution of the posterior. In addition, each particle is assigned a weight that determines how likely it is to be the true state, given the received measurements. The weights are updated every time a new measurement is received – the weights are increased for particles closer to the measurement.

Formally, let $\mathcal{X}_k = \{x_k^{[1]}, \ldots, x_k^{[M]}\}$ denote the set of M particles at time k; each $x_k^{[i]}$ has been drawn from $p_{k|k}$, i.e., the posterior distribution of the state at time k. Each particle $x_k^{[i]}$ has a weight w_i that represents the probability that the true state is equal to $x_k^{[i]}$ based on the empirical particle distribution. Thus, $\mathcal{X}_k$, together with all weights, represents the empirical distribution at time k. For instance, if one were to compute an estimate of the state, it is done as follows:

$$\hat{x}_k = \sum_{i=1}^{M} w_i x_k^{[i]}. \tag{13}$$

Other moments can be computed similarly.

The particle filter algorithm is summarized in Algorithm 1. In order to compute the posterior $p_{k+1|k+1}$, the particle filter also has predict and update stages. For each particle $x_k^{[i]}$, the prediction step consists of drawing a sample from $p(x_{k+1} \mid x_k^{[i]})$, which represents the predicted "estimate" of that particle. Upon receiving a new measurement, each particle's weight is updated as follows:

$$w_i^* = w_i p(y_{k+1} \mid x_{k+1}^{[i]}), \tag{14}$$

Algorithm 1 Particle Filter Algorithm

Input: A set $\mathcal{X}_k$ of M particles, a set $\mathcal{W}$ of weights, and the new measurement y_{k+1}.

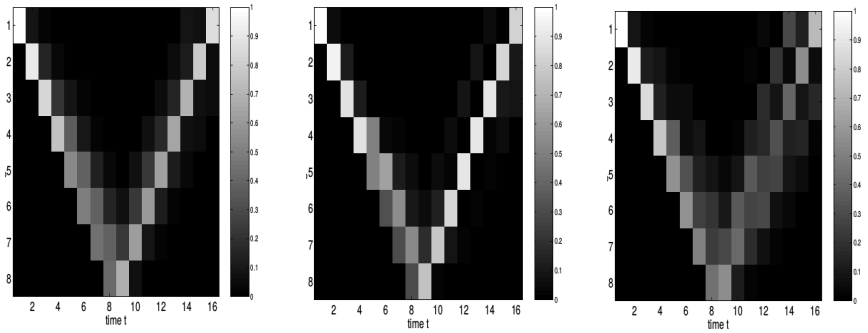
$\bar{\mathcal{X}}_{k+1} \leftarrow \emptyset$
$\bar{\mathcal{W}}_{k+1} \leftarrow \emptyset$
for $i = 1$ to M **do**
    $x_{k+1}^{[i]} \leftarrow \texttt{sample}(p(x_{k+1} \mid x_k^{[i]}))$
    $\texttt{add}(x_{k+1}^{[i]}, \bar{\mathcal{X}}_{k+1})$
    $w_i^* \leftarrow w_i p(y_{k+1} \mid x_{k+1}^{[i]})$
    $\texttt{add}(w_i^*, \bar{\mathcal{W}}_{k+1})$
end for
$\mathcal{X}_{k+1} \leftarrow \emptyset$
$\mathcal{W}_{k+1} \leftarrow \emptyset$
for $i = 1$ to M **do**
    draw integer p with probability $\propto w_p^*$
    $\texttt{add}(x_{k+1}^{[p]}, \mathcal{X}_{k+1})$
    $\texttt{add}(w_p^*, \mathcal{W}_{k+1})$
end for
return $(\mathcal{X}_{k+1}, \mathcal{W}_{k+1})$

i.e., the updated weight is larger when the latest measurement is more likely to have come from $x_{k+1}^{[i]}$. Weights are then renormalized so that they sum to 1 again.

Note that an extra step may be required in order to ensure that the particles have been generated from $p_{k+1|k+1}$, following the update with y_{k+1}. This is achieved via resampling – M new particles are sampled with replacement from the set $\mathcal{X}_{k+1}$, where each particle $x_{k+1}^{[i]}$ is selected with probability equal to its weight w_i^*.

The theoretical motivation for the particle filter is the law of large numbers and similar results about distribution convergence (e.g., the Glivenko-Cantelli theorem). These results say that when a large number of samples are drawn from a probability distribution, then the empirical distribution approaches the true distribution. The resampling step ensures that the resampled particles are drawn from the true posterior distribution and will approximate it given enough particles.

On the other hand, not many results exist about any finite set of particles. In general, it is difficult to estimate in advance how many particles are needed for any specific application. Furthermore, practice shows that the number of particles needed in order to achieve good estimation grows very quickly with the state dimension and can lead to particle deprivation problems in high-dimensional spaces [37]. That is why particle filters have been mostly applied to systems with a reasonably low number of dimensions, e.g., robotics; applications are discussed in Section 5.3.



(a) Exact Bayesian Inference.

(b) Rao-Blackwellised particle filter with 50 particles.

(c) Boyen Koller algorithm [4].

Figure 3: Evaluation of the Rao-Blackwellised particle filter on SLAM [9]. The grid number is shown in the vertical axis, whereas the horizontal axis shows elapsed time.

5.2 Rao-Blackwellised Particle Filter

An important extension of the particle filter is the so-called Rao-Blackwellised particle filter [10]. Motivated by the particle deprivation problem described above, it assumes that the state can be separated into two sets, one that can be filtered in closed-form (e.g., with a Kalman filter) and one that is estimated with the classical particle filter. When possible, this separation results in a smaller number of particles necessary for good estimation since it effectively reduces the dimensionality of the state estimated by the particle filter.

Formally, the Rao-Blackwellised particle filter works by splitting the state into two substates, i.e., $x_k = (x_{1,k}, x_{2,k})$. Then, the posterior is expressed by using Bayes rule:

$$p(x_{1,k}, x_{2,k} \mid y_1, \ldots, y_k) = p(x_{1,k} \mid x_{2,k}, y_1, \ldots, y_k) p(x_{2,k} \mid y_1, \ldots, y_k). \quad (15)$$

Thus, if $p(x_{1,k} \mid x_{2,k}, y_1, \ldots, y_k)$ can be computed in closed-form (e.g., if $x_{1,k}$ can take at most finitely many values), then only the second part, $p(x_{2,k} \mid y_1, \ldots, y_k)$, needs to be computed using the particle filter.

Similar convergence properties exist for the Rao-Blackwellised particle filter as for the classical particle filter. While the same caveat applies about any finite number of particles, the gains in the number of particles can be significant (i.e., orders of magnitude fewer particles) when the state is divided appropriately [10]. The next subsection discusses one successful application of this filter.

5.2.1 Evaluation

The Rao-Blackwellised particle filter has been applied successfully to the SLAM problem, where the separation of the state into a map and the robot's location seems natural [14]. In the original application [9], a robot moves in a two-dimensional grid environment with 8 cells. Each cell has a value of 1 or 0

depending on whether it contains an obstacle or not, respectively. The robot gets a measurement of its location plus a (possibly wrong) measurement of the value of each cell; the robot's goal is to estimate both its location and the map's obstacles.

The Rao-Blackwellised particle filter can be applied to this problem because it is assumed that cell value measurements are conditionally independent given the robot's location. In particular, by assuming that the probability of a fixed cell c_i being 1 or 0 is independent of the probability of any other cell c_j being 1 or 0, conditioned on the measurements and on the robot's position, it is possible to compute the first probability in (15) in closed-form. Thus only the robot's location is estimated using particles.

Using this state separation, the filter needs only 50 particles to achieve good estimation accuracy. The results are shown in Figure 3. It shows the robot's belief at each time step as it moves through the grid. The figure shows that the Rao-Blackwellised particle filter with only 50 particles (Figure 3b) performs almost as well as exact Bayesian inference (Figure 3a), whereas the Boyen Koller algorithm [4] (Figure 3c) performs much worse and gets confused at a few steps.[3]

Therefore, the Rao-Blackwellised particle filter is a great approach for solving non-linear estimation problems with a clear separation of the state into substates.

5.3 Applications

The particle filter has been applied in multiple areas, though by far the most dominant is robotics. Mainly applied to SLAM, it has been used in applications with both ground [8] and aerial [15] vehicles. Furthermore, it has been used in target tracking [22], hydrologic data assimilation [29], acoustic source localization [38] and others.

6 Set Membership Filters

As mentioned above, particle filters are useful in many non-linear system scenarios, yet they do not achieve very good performance in high-dimensional states. While Kalman filter extensions for non-linear systems do not suffer from the curse of dimensionality, they underperform when the analyzed systems are highly non-linear, thus leading to linearizations around wrong points and high estimation errors. As another alternative to filtering in non-linear systems, set membership methods have been proposed.

At a high level, set membership filtering works by propagating a set of all possible values for the true state. Such a set is usually achieved by assuming that both process and measurement noise are bounded. In this paper, we describe one of the most popular set membership approaches in the literature [27].

[3]The Boyen Koller algorithm is an inference algorithm for dynamic Bayesian networks. It is used for comparison in the original Rao-Blackwellised particle filter paper [9].

6.1 Problem Formulation

In this problem space, we assume a non-linear system with a non-linear observational model of the form:

$$x_{k+1} = f(x_k) + w_k$$
$$y_k = g(x_k) + v_k,$$

where the only difference from the probabilistic model is that no assumptions are made about the noise distribution, except that it is bounded, i.e., $\|w_k\| \leq \alpha$, $\|v_k\| \leq \delta$ for some positive α and δ.

As discussed in Section 3, we assume that system dynamics f is bounded in some way and hence belongs to the class of functions $\mathcal{K}$ (finding a suitable such set is also part of the problem solution); this allows one to be able to reason about all possible dynamics. The problem is to develop a filter h that minimizes the worst-case error as defined in Section 3:

$$e_{wc}(h, y_1, \ldots, y_k) = \sup_{\hat{h} \in \mathcal{K}} \sup_{\tilde{y}_i \in \mathcal{B}_\delta(y_i)} \|h(y_1, \ldots, y_k) - \hat{h}(\tilde{y}_1, \ldots, \tilde{y}_k)\|. \tag{16}$$

6.2 Approach

In this section we describe the approach taken by Milanese et al. [26, 27]. Rather than developing a recursive filter similar to the ones shown in the previous two sections, they employ a regression approach, i.e., they investigate the function f_x mapping all past measurements to the current state. The motivation for this approach is the following Theorem (modified and adapted to the notation of this paper):

Theorem 4 ([27])**.** *Assume that $v_k = 0$ and $w_k = 0$ for all k. If the system is observable, then there exists a function f_x such that*

$$x_k = f_x(\varphi_k)$$
$$\varphi_k = [y_1, \ldots, y_k].$$

Remark. *In the interest of space, the notion of observability is not introduced in this paper because it does not affect the analysis. In addition, every unobservable system can be reduced to an observable system, so we assume all systems considered in this work are observable.*

What Theorem 4 suggests is that if the system operates nominally and no noise is observed, then the current state can be expressed as a function f_x of all available measurements. Thus, when we consider the available noisy measurements, we have the following relation between the states and the measurements:

$$y_k = f_x(\varphi_k) + d_k,$$

where d_k is a disturbance term that accounts for the difference between nominal behavior and the noise. Similar to v_k and w_k, d_k is also bounded, i.e., $\|d_k\| < \epsilon$ for all k.

Thus, one way of evaluating the accuracy of any estimator would be to compare it with f_x. However, f_x is not known in general, and no information can be obtained about it unless further assumptions are made. In particular, the authors assume that f_x is continuously differentiable with a bounded derivative, i.e.,

$$f_x \in \mathcal{K} := \{f \in C^1 \mid \|f'(\phi_k)\| \leq \gamma\},$$

where C^1 is the class of all continuously differentiable functions. Thus, in order to quantify the accuracy of an estimator, one only needs to compare it with other functions in the set $\mathcal{K}$. Note that the set of possible estimators is further restricted by the available data; in particular, any estimator must not have empirical error more than ϵ. The set of feasible estimators $\mathcal{F}$ is then restricted as follows:

$$\mathcal{F} = \{f \in \mathcal{K} \mid \|y_k - f(\varphi_k)\| \leq \epsilon\}.$$

Thus, given the assumptions about f_x, $\mathcal{F}$ represents the smallest set of functions in $\mathcal{K}$ that have not been invalidated by the data.

Having defined $\mathcal{F}$, it is now possible to define the worst-case error for any estimator h:

$$e_{wc}(h, y_1, \ldots, y_k) = \sup_{\hat{h} \in \mathcal{F}} \sup_{\tilde{y}_i \in \mathcal{B}_\delta(y_i)} \|h(y_1, \ldots, y_k) - \hat{h}(\tilde{y}_1, \ldots, \tilde{y}_k)\|,$$

where $B_\delta(y_i)$ accounts for the fact that each measurement is at most δ away from the true state (i.e., measurement noise is bounded by δ as mentioned in the problem formulation). Given this error, the problem is now to find an optimal estimator that minimizes it.

The first step in finding such an estimator is to find tight upper and lower bounds given any set of measurements. The authors show that the following two functions are in fact tight upper and lower bounds on the estimate:

$$\bar{f}(\tilde{y}) = \min_{k=1,\ldots,M} (y_k + \epsilon + \gamma\|\tilde{y} - y_k\|)$$
$$\underline{f}(\tilde{y}) = \max_{k=1,\ldots,M} (y_k - \epsilon - \gamma\|\tilde{y} - y_k\|),$$

where M is a any fixed number of steps and y_k are the received measurements as before. Note that $\tilde{y}$ is just a function input, i.e., the above two functions are defined for all possible "measurements" regardless if they are actually possible given the system in consideration. The intuition behind $\bar{f}$ is that at any given step k, the true state cannot be farther than the received measurement plus the measurement error; the last term also incorporates process noise; since this relation is true for any k, by taking the minimum over all k, we obtain a tight upper bound. A similar intuition exists for $\underline{f}$.

Theorem 5 ([26]). *The functions $\bar{f}$ and $\underline{f}$ are optimal bounds, i.e., the interval $[\underline{f}, \bar{f}]$ is the smallest interval guaranteed to contain the optimal estimate.*

Given these two functions, we are now ready to develop the optimal estimator.

Theorem 6 ([27]). *The function $f_c = 1/2[\bar{f}+\underline{f}]$ is an almost optimal estimator, i.e.,*

$$e_{wc}(f_c, y_1, \ldots, y_k) \leq (1/2)[\bar{f}(y_k) - \underline{f}(y_k)] + \gamma\delta.$$

Remark. *The term "almost optimal" is precisely defined by Milanese et al. [27]. It is a relaxed, yet useful, definition of optimality that is easier to achieve by certain algorithms. We omit the definition in the interest of space.*

In summary, the set membership approach works by constructing the functions $\bar{f}$ and $\underline{f}$ from existing data, and then using them to perform estimation given future measurements. Several parameters need to be selected in order for these functions to be well defined, namely the bounds on the process and measurement noise, as well as the bounds on the error and on the derivative of the true regression function f_x.

6.3 Evaluation

To illustrate the advantages of the set membership filter, the authors evaluated it on a highly non-linear system, namely the Lorenz chaotic system:

$$\begin{aligned}
x_{1,k+1} &= (1-\tau\sigma)x_{1,k} + \tau\sigma x_{2,k} \\
x_{2,k+1} &= (1-\tau)x_{2,k} - \tau x_{1,k}x_{3,k} + \tau\rho x_{1,k} \\
x_{3,k+1} &= x_{3,k} + \tau x_{1,k}x_{2,k} - \tau\beta x_{3,k} \\
y_{1,k} &= x_{1,k} \\
y_{2,k} &= x_{2,k}x_{3,k},
\end{aligned}$$

where $\tau = 0.01, \sigma = 10, \rho = 28, \beta = 2.6667$. This system has both non-linear dynamics and a non-linear observation model, thus making it difficult to analyze using a Kalman filter. The goal for the filter is to compute an estimate of $x_{2,k}x_{3,k}$.

To evaluate the performance of the set membership filter, it is compared with an extended Kalman filter. The estimation results are shown in Figure 4. As can be seen in the Figure, the set membership estimates are much less susceptible to the non-linearities in the system; on the other hand, the extended Kalman filter suffers greatly due to linearizing around the wrong point and not capturing some of the sharp changes in dynamics.

Thus, when initialized properly and with correct parameters, the set membership approach can be a powerful method for performing state estimation.

6.4 Other Approaches

Other approaches to set membership filtering exist in the literature as well, depending on how the process/measurement noise is bounded. In particular, if

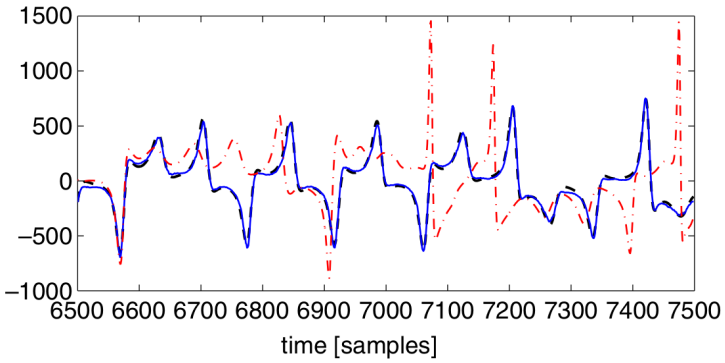


Figure 4: Evaluation of the set membership filter on the Lorenz system. Dashed (black): true signal; dot-dashed (red): extended Kalman filter estimate; continuous (blue): set membership filter estimate.

energy constraints on the noise are assumed (i.e., ellipsoids), it is possible to derive a recursive filter that has a similar form to the classical Kalman filter [3, 34]. In a more recent approach, researchers have explored the norm-bounded uncertainty problem and have applied convex optimization techniques in order to obtain an optimal estimate [11, 40]. Finally, recursive schemes for constructing ellipsoidal sets described by sum quadratic constraints have been developed [33].

Another important subfield of set membership methods is sensor fusion [25]. In this domain, the system in consideration is assumed to have multiple sensors measuring the same variable. By fusing all measurements, the system is made robust to sensor failures or attacks. Sensors are assumed to provide a bounded set of values; these sets are intervals in the one-dimensional case [18, 20, 25] and can be generalized to different shapes in multidimensional cases (e.g., hyperrectangles [5] or polyhedra [19]) depending on the problem. These sets are then combined in order to produce a minimal set that is guaranteed to contain the true value (e.g., by assuming that at least some number of measurements contain the true value). Sensor fusion has been successfully applied in fault/attack detection problems [18, 25] as well as in robust state estimation [32].

7 Discussion and Conclusion

This paper presented three classes of state estimation filtering techniques that cover a wide variety of applications observed in practice. These techniques were developed with different assumptions and applications in mind; hence, providing a fair comparison is not straightforward. In this section, we provide general observations and comments; however, specific applications may be found in support of either of these methods.

First, we note that if the system in consideration is linear, then a Kalman

filter is most likely the best approach. Even if the noise is not Gaussian (or if even if it is hard to identify any probability distribution), the Kalman filter still has strong theoretical guarantees. In addition, its numerous successful applications over the years have cofirmed its place as the most popular filtering algorithm.

If the system's model is non-linear, however, then the choice of filter is made more difficult. The extended Kalman filter (EKF) is also a popular choice in this case; it is especially powerful in systems that are mildly non-linear, e.g., a differential drive ground vehicle. However, in highly non-linear systems (e.g., quadrotors), the EKF may diverge due to linearizing around a wrong estimate. Furthermore, the EKF is computationally expensive in high-dimensional spaces as inverting matrices with more than a dozen states may be too slow on a microprocessor.

In cases where the EKF performs poorly, the next logical choice is the particle filter. Particle filters have been shown to outperform the EKF in highly non-linear systems, both in terms of accuracy and computational cost. Another appealing property of particle filters is that their accuracy can always be enhanced given more particles; thus, the advancement of modern computers is yet another argument in favor of the particle filter. On the other hand, particle filters suffer from the curse of dimensionality and perform poorly in high-dimensional spaces. In particular, the number of required particles grows very quickly with the dimensionality of the state, thus making it prohibitive to perform accurate estimation in high-dimensional spaces.

Finally, if neither of the above approaches leads to acceptable results, a set membership approach should be considered. As shown in Section 6, such an approach can greatly outperform an EKF in highly non-linear systems. The challenge with set membership approaches is that multiple parameters must be selected in order for the algorithm to perform well. The selected bounds greatly affect the filter's performance – if too small values are chosen, then the filter might diverge; if too large values are selected, then the estimates might be biased.

In conclusion, which method is selected depends first and foremost on the system model and on the process and measurement noise. A choice must be made between a very precise, but difficult to analyze, model and a model that is simple, but not very descriptive of the true system behavior. Once a model is determined and the noise is established accordingly, then a filtering technique can be selected based on the above observations.

References

[1] D. L. Alspach and H. W. Sorenson. Nonlinear bayesian estimation using gaussian sum approximations. *Automatic Control, IEEE Transactions on*, 17(4):439–448, 1972.

[2] N. Atanasov, R. Tron, V. M. Preciado, and G. J. Pappas. Joint estimation and localization in sensor networks. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 6875–6882. IEEE, 2014.

[3] D. P. Bertsekas and I. B. Rhodes. Recursive state estimation for a set-membership description of uncertainty. *Automatic Control, IEEE Transactions on*, 16(2):117–128, 1971.

[4] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 33–42. Morgan Kaufmann Publishers Inc., 1998.

[5] P. Chew and K. Marzullo. Masking failures of multidimensional sensors. In *SRDS'91: Proc. 10th Symposium on Reliable Distributed Systems*, pages 32–41, 1991.

[6] N. D. Crump. A kalman filter approach to the deconvolution of seismic signals. *Geophysics*, 39(1):1–13, 1974.

[7] P. Del Moral. Non-linear filtering: interacting particle resolution. *Markov processes and related fields*, 2(4):555–581, 1996.

[8] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 2, pages 1322–1328. IEEE, 1999.

[9] A. Doucet, N. De Freitas, K. Murphy, and S. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.

[10] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

[11] L. El Ghaoui and G. Calafiore. Robust filtering for discrete-time systems with bounded noise and parametric uncertainty. *Automatic Control, IEEE Transactions on*, 46(7):1084–1089, 2001.

[12] N. J. Gordon, D. J. Salmond, and A. F. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113. IET, 1993.

[13] M. S. Grewal and A. P. Andrews. Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]. *Control Systems, IEEE*, 30(3):69–78, 2010.

[14] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1):34–46, 2007.

[15] S. Grzonka, G. Grisetti, and W. Burgard. Towards a navigation system for autonomous indoor flying. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 2878–2883. IEEE, 2009.

[16] V. M. Hernandez Bennetts, A. J. Lilienthal, A. A. Khaliq, V. Pomareda Sese, and M. Trincavelli. Towards real-world gas distribution mapping and leak localization using a mobile robot with 3d and remote gas sensing capabilities. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2335–2340. IEEE, 2013.

[17] M. Huber. *Nonlinear Gaussian Filtering: Theory, Algorithms, and Applications*, volume 19. KIT Scientific Publishing, 2015.

[18] R. Ivanov, M. Pajic, and I. Lee. Attack-resilient sensor fusion. In *DATE'14: Design, Automation and Test in Europe*, 2014.

[19] R. Ivanov, M. Pajic, and I. Lee. Resilient multidimensional sensor fusion using measurement history. In *Proceedings of the 3rd international conference on High confidence networked systems*, pages 1–10. ACM, 2014.

[20] D. N. Jayasimha. Fault tolerance in a multisensor environment. In *SRDS'94: Proc. 13th Symposium on Reliable Distributed Systems*, pages 2–11, 1994.

[21] R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[22] Z. Khan, T. Balch, and F. Dellaert. An mcmc-based particle filter for tracking multiple interacting targets. In *Computer Vision-ECCV 2004*, pages 279–290. Springer, 2004.

[23] T. D. Larsen, M. Bak, N. A. Andersen, and O. Ravn. Location estimation for autonomously guided vehicle using an augmented kalman filter to autocalibrate the odometry. In *FUSION98 Spie Conference*. Citeseer, 1998.

[24] R. Mahler. *Statistical Multisource-Multitarget Information Fusion*. Artech House, Inc., 2007.

[25] K. Marzullo. Tolerating failures of continuous-valued sensors. *ACM Trans. Comput. Syst.*, 8(4):284–304, 1990.

[26] M. Milanese and C. Novara. Set membership identification of nonlinear systems. *Automatica*, 40(6):957–975, 2004.

[27] M. Milanese, C. Novara, K. Hsu, and K. Poolla. The filter design from data (fd2) problem: Nonlinear set membership approach. *Automatica*, 45(10):2350–2357, 2009.

[28] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, et al. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *AAAI/IAAI*, pages 593–598, 2002.

[29] H. Moradkhani, K.-L. Hsu, H. Gupta, and S. Sorooshian. Uncertainty assessment of hydrologic model states and parameters: Sequential data assimilation using the particle filter. *Water Resources Research*, 41(5), 2005.

[30] R. Olfati-Saber. Distributed kalman filtering for sensor networks. In *Decision and Control, 2007 46th IEEE Conference on*, pages 5492–5498. IEEE, 2007.

[31] R. Olfati-Saber. Kalman-consensus filter: Optimality, stability, and performance. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 7036–7042. IEEE, 2009.

[32] J. Park, R. Ivanov, J. Weimer, M. Pajic, I. Lee, and S. Son. Security of cyber-physical systems in the presence of transient sensor faults. 2015. Submitted.

[33] W. Ra, S. Jin, and J. Park. Set-valued estimation approach to recursive robust h filtering. *IEE Proceedings-Control Theory and Applications*, 151(6):773–782, 2004.

[34] F. C. Schweppe. Recursive state estimation: unknown but bounded errors and system inputs. *Automatic Control, IEEE Transactions on*, 13(1):22–28, 1968.

[35] D. Simon and D. L. Simon. Constrained kalman filtering via density function truncation for turbofan engine health estimation. *International Journal of Systems Science*, 41(2):159–171, 2010.

[36] B. L. Stevens and F. L. Lewis. *Aircraft control and simulation*. John Wiley & Sons, 2003.

[37] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT press, 2005.

[38] D. B. Ward and R. C. Williamson. Particle filter beamforming for acoustic source localization in a reverberant environment. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 2, pages II–1777. IEEE, 2002.

[39] N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT press Cambridge, MA, 1949.

[40] F. Yang and Y. Li. Set-membership filtering for systems with sensor saturation. *Automatica*, 45(8):1896–1902, 2009.