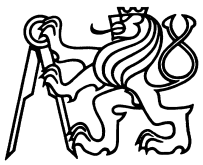
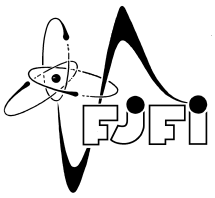


Czech Technical University in **Prague**
Faculty of **Nuclear Sciences and Physical Engineering**



Distributed Kalman filtration under unknown spatially heterogeneous noise

Master's Thesis

Author: Bc. Daniel Hnyk

Supervisor: Ing. Kamil Dedecius, Ph.D.

Academic year: 2017/2018

Acknowledgment:

I would like to thank my thesis supervisor Mr. Dedecius for a constant support and leadership through the whole process. It was not just his expertise and knowledge which I had a chance to experience. But it was also an instant and high-quality feedback on my progress, always provided in a respectful and helpful way. I hope that it led to a valuable contribution to science and it also made working on this thesis enjoyable.

Declaration:

I hereby certify that this text represents my own work and that all used sources and materials are listed in the bibliography.

Prague, Sunday 29th April, 2018 Daniel Hnyk

Title: Distributed Kalman filtration under unknown spatially heterogeneous noise

Author: Bc. Daniel Hnyk

Branch of *study:* Mathematical Informatics

Abstract: Combination of adaptive filters in time-varying noisy environment and diffusion networks is not an easy task. A generic algorithm synthesizing state-of-the-art Variational Bayes techniques, Adaptive Kalman Filtration and diffusion networks is proposed and leveraging the exponential family form representation and conjugate priors, a concise derivation is carried out. The proposed algorithm is applied to the problem of collaborative estimation of a trajectory of a target tracked by a group of spatially distributed interconnected agents influenced by heterogeneous noise. It is shown, that cooperation significantly improves tracking performance of the whole agent network.

Key words: Bayesian inference, Kalman filter, time-varying noise covariance matrices, diffusion network, adaptation, state estimation

Contents

Introduction

A rapid development in the information technology field over the last two decades boosted affordability of cheap, yet powerful devices capable of carrying out complex computations. Variety of available sensors, which these devices can be equipped with, is overwhelming. Omnipresent smartphones or cars contain hundreds of sensors constantly collecting data and deciding on life-saving or life-threatening actions. Surveillance systems, composed of hundreds or even thousands of interconnected sensors keep national security in place, always prepare to act in seconds based on the latest data. The so-called Internet of Things, many small computers connected to each other is only another demonstration of this phenomenon. At the same time, the environments where these devices operate often suffer from the imperfections of their measurements or computations. These may be caused by, e.g., misalignment of their models with the reality or by noisy random fluctuations in their operations.

For the sake of getting valuable insights from the collected data of these noisy data collections and evaluation networks, efficient processing algorithms must be developed. There is a long history of research in so-called consensus algorithms as well as in the noise filtering methods. While the former try to solve fundamental problems in the area of distributed computing and multi-agent systems of agreeing on quantity of interest, the latter deal with the omnipresent noise in the received data. What is worse, many methods are sensitive to a certain type of noise, making them inappropriate in some use cases.

Not being able to effectively filter noise from the data makes the data less useful than it otherwise could be. The same is true in the cases when the estimation redundancy is not exploited (for instance, averaging over data from multiple sensors). One of the notable approaches trying to solve this problem is Distributive Kalman Filtering, an extension of a massively popular and heavily used filter since its formulation in 1960. This work is concerned with one particular instance of the Distributive Kalman Filtering, using state-of-the-art techniques and the latest research in probability theory. It combines a specific probabilistic variant of the classic Kalman Filter with a recently developed so-called parameter diffusion. Both techniques stem from the fact that they can be fully expressed in Bayesian probability terms and it seems to that they can be an excellent candidate as a solution to the problem of combining information from multiple agents in the noisy environment. The following pages are describing the development of such algorithm and explain under which conditions it may be a suitable solution. Additionally, real implementation in Python language is used to run various simulations to validate the theoretical predictions.

The work is divided into a theoretical and a practical part. The former states all the necessary theoretical background for the latter. In the first chapter, the Bayesian probability is introduced as a building block for Variational Bayes inference. This is a general technique which serves as the crux for the proposed algorithm internals. The second chapter is concerned with Kalman filtering and its distributed extension. A general introduction with brief applications overview is followed by mathematical formulation and discussion of its key properties. The third chapter is a connection of these components – Variational Bayes and Distributed Kalman filter-

ing. The derivation of the mathematical formulation using exponential family form is performed and in the end, a general, step-by-step algorithm is outlined. The final chapter containing the results firstly comments on implementation details including the development ecosystem. Subsequently, simulations and comparison of different variants are carried out, evaluating the algorithm performance, explaining the observed behavior and pinpointing its properties, features, and disadvantages. Each of the chapters contains an introductory text outlining what is going to be presented and if necessary, establishes the minimum requirements.

Chapter 1

Bayesian probability

In this chapter, I summarize basics and preliminaries of Bayesian interpretation of probability and shortly compare it to the frequentist approach. The reader is expected to know basic concepts from probability such as a probability event, space, measure, product and sum rule, conditional and joint probability. Sufficient description for all of these concepts can be found in [1].

1.1 Probability interpretations

As it is the case with some other branches of mathematics, probability theory has multiple interpretations with their own strengths and weaknesses. In this particular case, it is a subject of dispute if the term "interpretations" should be used since there is no single formal system "probability". For example, some of the leading interpretations fail to satisfy the most common Kolmogorov's axiomatization and they still prove themselves useful[2]. Another axiomatizations have been proposed for different interpretations [3] and some of the leading interpretations such as [1] do not even derive from mathematical axioms, but rather somewhat vague statements such as:

1. representations of plausibility are to be given by real numbers
2. plausibilities are to be in qualitative agreement with common sense
3. the plausibilities are to be "consistent", in the sense that anyone with the same information would assign the same real numbers to the plausibilities.

Interpretations do not differ only in the underlying set of axioms, but also in their practical usage or their epistemological status. Mathematicians, as well as philosophers, have been tackling these issues at least for over five centuries.

It is hard to fully appreciate and understand main concepts and advantages of Bayesian reasoning if one is not familiar with other interpretations, such as the frequentist one. Its full derivation can be found in [4] and here we cover just the basics arising from the comparison and parts relevant for the following pages.

The main idea of Bayesian interpretation is its subjective nature, where there is no such thing as the *real* probability in the world, but the probabilities are in the actor's mind. Probability then represents agent's *state* of *knowledge* or *degree* of *belief* [5, 6, 7]. This is not a negligible formal difference – it has important consequences, such as the fact that all agents with different state of knowledge can assign different probabilities to the same event. In Bayesian framework, one explicitly convey uncertainty in his models and objects (such as random variables).

Another important aspect worth mentioning is also the fact that Bayes interpretation allows doing predictions for events which have not occurred yet, where frequentists are (formaly) unable

to do anything. This property is also connected to a fact that one can do partial (*online*) updates based on the new evidence.

1.2 Principles of Bayesian probability

The core idea of Bayesian interpretation is the famous Bayes theorem (1.1), which is easily obtained from the definition of conditional probability [8].

Theorem 1. Let p be *a* probability measure, $\mathbf{A}$ and $\mathbf{B}$ are events, $p(\mathbf{B}) \neq 0$. Then

$$p(\mathbf{A}|\mathbf{B}) = p(\mathbf{A}) \cdot \frac{p(\mathbf{B}|\mathbf{A})}{p(\mathbf{B})} \tag{1.1}$$

where:

1. $p(\mathbf{A})$ and $p(\mathbf{B})$ are the probabilities of observing A and B independently.
2. $p(\mathbf{A}|\mathbf{B})$ is the probability of observing event A given B.
3. $p(\mathbf{B}|\mathbf{A})$ is the probability of observing event B given A. This factor is called **likelihood**.

To relate this simple formula to the previous section, consider a scenario where one wants to use Bayesian framework for *inference.* Suppose that $\mathbf{A}$ is a hypothesis of which the probability can be affected by some evidence. Let $\mathbf{B}$ be the evidence for this hypothesis. $p(\mathbf{A})$ in formula (1.1) can then be seen as a **prior probability** – a probability one would assign to $\mathbf{A}$ is true *before* observing the evidence. Factor $p(\mathbf{B}|\mathbf{A})$, the *likelihood*, represents how likely we expect to observe $\mathbf{B}$ *given* that $\mathbf{A}$ is true. $p(\mathbf{B})$ is the same for all possible hypothesis and serves as a normalizing factor. That is the reason why it is often left out in many computations where only relative comparison is needed. And finally, $p(\mathbf{A}|\mathbf{B})$ is the final **posterior probability** which is the factor one usually cares about the most. It is the probability of $\mathbf{A}$ after observing $\mathbf{B}$.

A factor ${}^{p(\mathbf{B}|\mathbf{A})}/_{p(\mathbf{B})}$ has also quite useful interpretation and that is the overall impact of evidence $\mathbf{B}$ on the probability of $\mathbf{A}$.

1.3 Statistical model

Chosen interpretation has a nontrivial influence on what such concept as a *statistical model* is and how a *statistical inference* is performed. We will follow traditional approach taken in [9, 10] and extend it by some necessary concepts from [11].

A statistical model is a pair of (S, M) where S is a set of possible observations (*sample space*) and M is a set of probability distributions on S. The distributions in M are expected to be approximately close to the *true* distribution which generates the data[1]. One of the distinctions which can be made to separate families of statistical models is how distributions in M are described:

1. **Parametric models**: in this case, the set M is parameterized by some parameter θ and can have values from *finite* parameter space $\Theta \subseteq \mathbb{R}^d$. In other words: $M = \{P_\theta | \theta \in \Theta\}$.

[1] A common aphorism in statistics, generally attributed to George Box, says: "All models are wrong, but some of them are useful".

2. **Nonparametric models**: this family is ipso facto a misnomer. Nonparametric models also have parameters, but they differ from parametric models in a way that the parameter space Θ is *infinite.* It is not fixed and can grow with the amount of data.

In Bayesian setting, parameters of the model have assigned some prior distributions. These distributions capture prior knowledge one has before the application of the model. For instance, this can be based on a previous research or reliable enough observations obtained beforehand. If there is no prior knowledge available, so-called non-informative priors are used instead. Uninformative is again a misnomer. What is meant by the name is the fact that they are not subjectively elicited. Fully Bayesian model is then a model in which *all* the parameters have some prior assigned. An interesting property of Bayesian approach arise – random variables, including latent variables, and parameters are treated in the exactly same way. That is again a major distinction from frequentist approach.

1.4 Statistical inference

The process of inferring properties of the data underlying distribution is called *statistical inference*[12]. Formally, it is justification for a restriction of the parameter space based on the data. An example is choosing a point estimate for a given parameter.

Unfortunately, for many real-world scenarios, the approach of directly inferring the properties is not possible. One of many reasons can be too high dimensionality of the space or the form of the posterior distribution may be too complex, let alone the fact that it does not even have to have the analytical solution[11]. In these cases, approximation methods have to be used instead.

1.4.1 Approximation methods

Using a breakdown from [11], approximation methods can be divided to two groups based on their stochastic or deterministic behavior. A notable example of the former is Markov chain Monte Carlo[11, (Chap.11)]. A representative approach of the latter is, for instance, variational inference described in detail in section Section 1.6. The main difference is the fact that stochastic schemes, *given infinite resources*, are guaranteed to get to the exact results and the approximation arise from the finite amount of those resources. On contrary, deterministic approximation schemes are based on *analytical approximations* to the posterior distribution. Hence, they are generally not able to generate exact results but may have different advantages such as tractability. Both approaches are then complementary to each other, one being useful for situation where the second is inapplicable and vice versa.

1.5 Conjugate distributions and exponential family

One of the remarkable properties of (1.1) is the fact that it can be used sequentially as an update based on the new piece of data. If one performs the update and obtains $p(\mathbf{A}|\mathbf{B})$, this posterior can be used in the next iteration as a prior repeatedly. Using the concept of conjugate priors exploits this remarkable property guaranteeing that one will get a tractable posterior after the each iteration.

In the most of the real-world applications, using Bayes theorem computationally means evaluating some integrals. For instance, to obtain the posterior, one must compute an integral $p(\mathbf{x}) = \int p(\mathbf{x}|\theta)p(\theta)d\theta$, where the integral is over the parameter space Θ. To make use of the theorem, one is then limited to problems where these integrals are tractable.

As already mentioned, this is done by using conjugate priors. Given the likelihood $p(\mathbf{x}|\theta)$, a family of prior distributions is chosen so that the integral $\int p(\mathbf{x}|\theta)p(\theta)d\theta$ is tractable and the result is in the same family as the prior. It is worth mentioning that these goals are usually in conflict. A tractable result is obtained if we just use some constant as the distribution family, but the update would not necessarily result in a constant. On the other hand, allowing all possible distributions leads to non-tractable solutions.

In practice, there is such a family containing many of the most commonly used distributions such as Gaussian or beta distribution. This family is called *exponential family* and guarantees that if the model belongs to an exponential family, then the conjugate prior always exists and hence the resulting posterior of the update is analytically tractable.

1.5.1 Formal definition

A formal definition of the above can be found in [13] and is restated here because the forms are going to be used further along in this work.

Definition 2. (Exponential family distribution). An exponential family distribution of a random variable y conditioned on x and with parameter Θ is a distribution of which the probability density function can be written in the form

$$p(y|x,\Theta) = \exp\Big\{\eta^T T - A(\eta) + k(x,y)\Big\} \tag{1.2}$$

where $\eta = \eta(\Theta)$ is the natural parameter of the exponential family distribution, $T := T(x,y)$ is a sufficient statistic encompassing all the information provided by x and y about the parameter Θ and

$$A(\eta) = \log \int \exp\Big\{\eta^T T + k(x,y)\Big\}\mathrm{d}x\mathrm{d}y \tag{1.3}$$

where the integral is over the space of x and y and $A(\eta)$ is a known log-partition function normalizing the density, and $k(x,y)$ is a known function independent of the parameter.

The form (1.2) is not unique since the product $\eta^T T$ can result in the same number when is the each factor multiplied by an appropriate constant.

Definition 3. (Conjugate prior distribution for Θ). Assume that a random variable y is conditioned by a variable x and obeys an exponential family distribution – model – with a parameter Θ. A prior distribution for Θ conjugate to the model is characterized by conjugate hyperparameters Ξ and Υ and has the probability density of the form

$$\pi(\Theta|\Xi,\Upsilon) = \exp\Big\{\eta^T\Xi + \Upsilon A(\eta) + l(\Xi,\Upsilon)\Big\} \tag{1.4}$$

where Ξ has the same size as $T, \Upsilon \in \mathbb{R}_+$ and $l(\Xi,\Upsilon)$ is a known function. The Ξ, Υ are called conjugate hyperparameters.

Substituting (1.2) and (1.4) into (1.1), the Bayesian update results into a simple summation.

Lemma 4. *(Bayesian update under conjugacy). Assume that a random variable y conditioned by a variable x has an exponential family distribution with a parameter Θ, estimated by means of a conjugate prior distribution. The Bayesian update is then equivalent to the update of conjugate hyperparameters*

$$\Xi_t = \Xi_{t-1} + T_t \tag{1.5}$$

$$\Upsilon_t = \Upsilon_{t-1} + 1 \tag{1.6}$$

The proof is omitted for its triviality and follows directly from the product of 1.2 and 1.4.

1.6 Variational Bayes inference

In this work, a family of deterministic approximation methods called *variational inference (*or *variational Bayes)* is used. For details, see [11, (p. 463)] and [14]. This technique is based on using the solution of an optimization problem to statistical inference. More specifically, finding an input which minimizes a specific functional. This method is guaranteed to get an exact result given some family of possible input functions over which one minimizes. The approximation arises from limiting the possible inputs, for instance, by considering only quadratic functions or, as is widely used and in this work, functions which factorize in a specific way.

Let $p(\mathbf{X}, \mathbf{Z})$ be Bayesian model, where all prior distributions are given. Let denote the set of all parameters and all latent variables $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ and the set of all observed variables $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$. The goal of the inference is to find the posterior distribution $p(\mathbf{Z}|\mathbf{X})$ and the distribution of the model evidence $p(\mathbf{X}) = \int p(\mathbf{Z}, \mathbf{X})\ \mathrm{d}\mathbf{Z}$ (the integral goes over the all latent variables), which is usually unavailable. Unfortunately, in many real-world scenarios, $p(\mathbf{Z}|\mathbf{X})$ is intractable. For example, one often cannot integrate all configurations of the hidden variables in the denominator (potentially infinite). That is where *variational lower bound* comes in. Instead of trying to compute $p(\mathbf{Z}|\mathbf{X})$ directly, we consider $q(\mathbf{Z})$ which is the as close approximation as possible to the former and has a convenient and tractable form – its expectations are computable. Furthermore, these approximate distributions can also have their own *variational parameters* which are considered to be in $\mathbf{Z}$ as well.

In the following lines, for the sake of readability, we omit arguments of distributions where possible. Most often it means $q := q(\mathbf{Z})$ and $p := p(\mathbf{Z}|\mathbf{X})$. As a measure of closeness between the approximate $q(\mathbf{Z})$ and $p(\mathbf{Z}|\mathbf{X})$, the Kullback-Leibler (KL) divergence is used:

$$\mathrm{KL}(q||p) = -\int q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})}\ \mathrm{d}\mathbf{Z} \tag{1.7}$$

and the integral goes again over the all latent variables. The KL divergence has the property that it is always non-negative $\mathrm{KL}(q||p) \geq 0$ and that $\mathrm{KL}(q||p) = 0$ if and only if $p(\mathbf{X}|\mathbf{Z}) = q(\mathbf{Z})$ almost everywhere. However, it is not a metric, since it does not satisfy the triangle inequality and is not symmetric.

The goal is then to solve the following optimization problem:

$$q(\mathbf{Z}) = \arg\min_{q(\mathbf{Z})} \mathrm{KL}(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X})). \tag{1.8}$$

The optimization (1.8) cannot be performed directly. To compute $\mathrm{KL}(q||p)$, the unknown evidence $p(\mathbf{X})$ is needed, as can be seen from the following derivation:

$$\mathrm{KL}(q||p(\mathbf{Z}|\mathbf{X})) = \mathbb{E}[\ln q] - \mathbb{E}[\ln p(\mathbf{Z}|\mathbf{X})] = \mathbb{E}[\ln q] - \mathbb{E}[\ln p(\mathbf{Z}, \mathbf{X})] + \ln p(\mathbf{X}).$$

Instead of trying to compute $\mathrm{KL}(q||p)$, we are going to optimize an alternative objective that is equivalent to $\mathrm{KL}(q||p)$ up to a constant. Let us then define *variational lower bound* as follows:

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})}\ \mathrm{d}\mathbf{Z} = \mathbb{E}[\ln p(\mathbf{Z}, \mathbf{X})] - \mathbb{E}[\ln q(\mathbf{Z})]. \tag{1.9}$$

It can be shown that $\mathcal{L}(q)$ has an important property being a lower bound of the log probability $\ln p(\mathbf{X}) \geq \mathcal{L}(q)$. Hence, when one wishes to maximize marginal $p(\mathbf{X})$, he can instead maximize its variational lower bound $\mathcal{L}(q)$.

Combining (1.9) and (1.7), the following relationship can be derived:

$$\ln p(\mathbf{X}) = \mathcal{L}(q) + \mathrm{KL}(q||p). \tag{1.10}$$

Since $p(\mathbf{X})$ does not depend on q, maximizing $\mathcal{L}(q)$ is possible and it is equivalent to minimizing $\mathrm{KL}(q||p)$.

In summary, because we do not know the true posterior distribution, we work with some family of distributions $q(\mathbf{Z})$ for which $\mathcal{L}(q)$ becomes tractable and search for the candidate which maximizes $\mathcal{L}(q)$ and through KL relates this candidate to the posterior. Obviously, the choice of the distribution family is critical here. While it needs to be tractable, it still needs to be flexible enough to provide sufficiently precise approximation, goals usually being in conflict. This is the part where the approximation arises in variational Bayes as has been discussed in 1.4.1.

1.6.1 Factorized distributions

A common choice for such a family of distributions described in the chapter Section 1.6 are *factorized distributions* c.f. *mean field theory* in physics[15]. The assumption on the family is quite simple. We treat each variable in $\mathbf{Z}$ as independent, in other words, the family **factorizes**:

$$q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}).$$

It is clear that this simplification is often not met in reality. Variables are dependent in the real world – it is often the reason why it is so hard to obtain the posterior distribution directly.

Notice that we do not require any specific form of $q_i(\mathbf{Z_i})$. Also, it is worth noting that this is *not* a model of the observed data – it is the variational lower bound and KL minimization problem, which connects this to the model and data.

1.7 Coordinate ascent mean-field variational

Coordinate ascent mean-field variational ($CAVI$) is a commonly used algorithm for solving optimization problem (1.8)[14][11].

1.7.1 Derivation

For the sake of readability, the expectation over all q distributions *except* q_i is denoted as $\mathbb{E}_{-i}[...]$. Similarly, $\mathbf{Z}_{\{-\mathbf{i}\}}$ means all latent variables except $\mathbf{Z_i}$.

Optimal $q_i^*(\mathbf{Z_i})$

First, consider the complete conditional of $\mathbf{Z_i}$, which is $p(\mathbf{Z_i}|\mathbf{Z}_{\{-\mathbf{i}\}}, \mathbf{X})$. If we fix all other variational factors $q_l(\mathbf{Z_l}), l \neq i$, the optimal $q_i^*(\mathbf{Z_i})$ satisfies the following

$$q_i^*(\mathbf{Z_i}) \propto \exp\{\mathbb{E}_{-i}[\ln p(\mathbf{Z_i}|\mathbf{Z_{-j}}, \mathbf{X})]\} \propto \exp\{\mathbb{E}_{-i}[\ln p(\mathbf{Z_i}, \mathbf{Z}_{\{-\mathbf{i}\}}, \mathbf{X})]\}, \tag{1.11}$$

wherein $q_{-i}(\mathbf{Z_{-i}}) = \prod_{l \neq i} q_l(\mathbf{Z_l})$.

Obtaining KL

Let us now rewrite $\mathcal{L}(q)$ in (1.9) as a function of $q_i(\mathbf{Z_i})$

$$\mathcal{L}(q_i) = \mathbb{E}_i[\mathbb{E}_{-i}[\ln p(\mathbf{Z_i}, \mathbf{Z_{-i}}, \mathbf{X})]] - \mathbb{E}_i[\ln q_i(\mathbf{Z_i})] + C \tag{1.12}$$

wherein the first term is just an iterated expectation and in the second we retain only product term containing $q_i(\mathbf{Z_i})$, while other parts can be moved to a constant C thanks to the use of a factorized distribution.

Now it is easy to see that (1.12) is a negative KL divergence between $q_i(\mathbf{Z_i})$ and $q_i^*(\mathbf{Z_i})$. That gives us all the necessary parts for the CAVI algorithm as described in [14]:

Algorithm 5. *CAVI*
Input: *A model* $p(\mathbf{X}, \mathbf{Z})$*, a dataset* $\mathbf{X}$
Output: *A variational density* $q(\mathbf{Z}) = \prod_{i=1}^m q_i(\mathbf{Z_i})$
Initialize: *Variational factors* $q_i(\mathbf{Z_i})$
while *the* $\mathcal{L}(q)$ *has not converged*
  for $i \in \{1, ..., m\}$ ***do***
    Set $q_i(\mathbf{Z_i}) \propto \exp\{\mathbb{E}_{-i}[\ln p(\mathbf{Z_i}, \mathbf{Z}_{\{-\mathbf{i}\}}, \mathbf{X})]\}$
  end
  Compute $\mathcal{L}(q) = \mathbb{E}[\ln p(\mathbf{Z}, \mathbf{X})] - \mathbb{E}[\ln q(\mathbf{Z})]$
end
return $q(\mathbf{Z})$

It is guaranteed that this algorithm converges to a local minimum.

1.8 Example

Let us demonstrate the above with an example from [11].

The model Consider the i.i.d dataset $\mathbf{X} = \{x_1, \cdots, x_N\}$ generated by (unknown) Gaussian distribution $\mathbf{X} \sim \mathcal{N}(\mu, \tau^{-1})$. The goal is to infer posterior $p(\mu, \tau|\mathbf{X})$ and the joint probability is

$$p(\mathbf{X}, \mu, \tau) = p(\mathbf{X}|\mu, \tau)p(\mu|\tau)p(\tau).$$

Factorized distribution Let us now approximate $p(\mu, \tau|\mathbf{X})$ by $q(\mu, \tau)$ with the assumption that q factorizes: $q(\mu, \tau) = q(\mu)q(\tau)$.

Conjugated priors The complete data likelihood is Gaussian distribution and hence in exponential family. We set a non-informative conjugate priors for the μ and τ as follows:

$$\mu \sim \mathcal{N}(\mu_0, (\lambda_0\tau)^{-1}),$$
$$\tau \sim \text{Gamma}(a_0, b_0).$$

where the hyperparameters $\mu_0, \lambda_0, a_0, b_0$ are initially set to some small positive number as in the usual Bayesian inference.

The form of $q(\mu)$ and $q(\tau)$ This is usually the hardest part of variational inference. The derivation of the following terms is based on (1.11) and in detail can be found in [11]. The derivations yields following for $q_\mu^*(\mu)$:

$$q_\mu^*(\mu) \sim \mathcal{N}(\mu \mid \mu_N, \lambda_N^{-1}),$$

where

$$
\begin{aligned}
\mu_N =& \frac{\lambda_0\mu_0 + N\bar{x}}{\lambda_0 + N}, \\
\lambda_N =& (\lambda_0 + N)\mathbb{E}_\tau[\tau] = (\lambda_0 + N)\frac{a_N}{b_N}, \\
\bar{x} =& \frac{1}{N}\sum_{n=1}^{N} x_n,
\end{aligned}
$$

and for $q_\tau^*(\tau)$:

$$
q_\tau^*(\tau) \sim \text{Gamma}(\tau \mid a_N, b_N),
$$

where

$$
\begin{aligned}
a_N &= a_0 + \frac{N+1}{2}, \\
b_N &= b_0 + \frac{1}{2}\mathbb{E}_\mu[\sum_{n=1}^{N}(x_n - \mu)^2 + \lambda_0(\mu - \mu_0)^2] \\
&= b_0 + \frac{1}{2}\left[(\lambda_0 + N)\left(\lambda_N^{-1} + \mu_N^2\right) - 2\left(\lambda_0\mu_0 + \sum_{n=1}^{N} x_n\right)\mu_N + \left(\sum_{n=1}^{N} x_n^2\right) + \lambda_0\mu_0^2\right].
\end{aligned}
$$

Perform updates based on CAVI Initially, compute $N, \sum_{n=1}^{N} x_n, \sum_{n=1}^{N} x_n^2$ and set $\lambda_{N,t}$ to some random value. Then iterate these steps until convergence:

1. plug in $\lambda_{N,t}$ to obtain $b_{N,t}$,
2. compute new $\lambda_{N,t+1}$ based on $b_{N,t}$ and all other terms needed.

Results After m iterations, we obtain new values of all parameters including hyperparameters. Hence we are able to approximate the posterior distribution. The joint probability is

$$
\begin{aligned}
p(\mathbf{X}, \mu, \tau) =& p(\mathbf{X}|\mu, \tau)p(\mu|\tau)p(\tau) \\
=& \prod_{n=1}^{N} \mathcal{N}(x|\mu_N, \tau_N^{-1})\mathcal{N}(\mu_N, (\lambda_N\tau_N)^{-1})\text{Gamma}(a_N, b_N).
\end{aligned}
$$

Chapter 2

Distributed Kalman filter

2.1 Introduction

Kalman filter (KF) is widely used adaptive filtering method firstly described in A *New Approach* to *Linear Filtering and Prediction Problems* by R.E. Kalman in 1960. The filter is a recursive solution to the discrete-time linear filtering problem [16] and it is one of the solutions to one of the most fundamental problems in control theory, the linear quadratic Gaussian control problem. Over the years, numerous variants and extensions have been derived from the original Kalman filter and this thesis is concerned with one of such extensions – Distributed Kalman Filtering. See Section 2.7 for a brief overview of used methods.

Kalman filter has been thoroughly studied over the years and also adopted by multiple industries, having a vast amount of applications wherever an estimation of the state of the process is needed and noise in prediction or observation is present. Because of its nature, it is commonly used in navigation and control of robots and vehicles[17], including spacecrafts[18]. It can be applied in time series analysis and hence is used in fields such as econometrics[19], economics[20] or signal processing[21]. Various applications can be found in neuroscience[22].

With respect to the current development of the so-called Internet of Things and development of cheap measurements sensors and microcomputers, Distributed Kalman Filtering (DKF) is becoming even more relevant. As one of the solutions to *sensor fusion* problem, it has been under an intensive research as can be seen in [23]. A network consisting of multiple sensors has many advantages over a single sensor scenario, such as robustness to noise, better field of view or, if designed appropriately, not having a single point of failure [24]. Applications range from wireless networks[25] to precision agriculture[26], military and civil surveillance, medical applications, nuclear hazard assessment and others[13].

The filter is defined by a set of relatively simple equations and allows for efficient online computation without having to know the whole history of data. It is the optimal linear filter under specific conditions, in detail described in Section 2.5.

2.2 Description

The basic Kalman filter is used to estimate the current state of a linear dynamic system in case of noisy measurements. The state can be described by various variables, such as position and velocity. To use a Kalman filter, one needs to specify system of interest in terms of the state-space equations, in other words:

1. control inputs, if any, such as sending a signal to steer a wheel,
2. dynamic model, such as physical laws of motion,

3. measurements, such as readings from sensors.

The filter is useful in situations where dynamic model nor measurements cannot be entirely trusted (otherwise one would not need to use it at all). It combines the information while allowing for some noise – uncertainty – during prediction and measurements. Both pieces of information are weighted by so called Kalman's gain and depending on its value the filter favors prediction over measurements and vice versa. The whole process is performed recursively, remembering the last estimate and covariance. Hence, it does not need the entire history, which makes it attractive where memory capacity is expensive. As a result of a less noisy estimates (hence a filter), better estimates are generated at each step.

Observable parameters coming from measurements do not have to be necessarily complete in a sense of describing the system in its entirety. It is a common case in which one observes only a subset of state variables, while the filter can still estimate the entire state.

The distributed extension is easy to describe. Instead of having only a single sensor, the system is measured by multiple of them and they are combined in a particular way. Despite its relatively simple high-level description, a multisensor network of sensors must deal with various non-trivial problems and the whole computation may be much more complicated than the non-distributed version. The complexity is based on degrees of freedom and selection of parameters of the network's behavior and topology. It also often means getting into a distinct research area of distributed data fusion[27] or consensus theory, which are out of the scope of this thesis.

2.3 Formal definition

2.3.1 Dynamical Model

Definition 6. Let $\mathbf{x}_k, \mathbf{x}_{k-1}, \mathbf{u}_k \in \mathbb{R}^n$ be the true state vectors at times $k, k-1$ and the control vector, respectively. $\mathbf{z}_k \in \mathbb{R}^m$ is the measurement vector, $\mathbf{F}_k, \mathbf{B}_k \in \mathbb{R}^{n\times m}$ the state transition matrix and the control matrix, and $\mathbf{H}_k \in \mathbb{R}^{m\times n}$ is the observation matrix. $\mathbf{w}_k \in \mathbb{R}^n$ and $\mathbf{v}_k \in \mathbb{R}^m$ are process noise and observation noise, respectively. The filter assumes that the state evolves from time $k-1$ to a new state in time k according to

$$\mathbf{x}_k = \mathbf{F}_k\mathbf{x}_{k-1} + \mathbf{B}_k\mathbf{u}_k + \mathbf{w}_k, \tag{2.1}$$

and then an observation is made as

$$\mathbf{z}_k{=}\mathbf{H}_k\mathbf{x}_k{+}\mathbf{v}_k. \tag{2.2}$$

In this work, the $\mathbf{B}_k\mathbf{u}_k$ part is omitted as is common in many scenarios where control input is not necessary. The process and observation noise $\mathbf{w}_k, \mathbf{v}_k$ are assumed to be zero mean (multivariate) Gaussian white noise with covariance matrices $\mathbf{Q}_k, \mathbf{R}_k$

$$\begin{aligned} \mathbf{w}_k &\sim \mathcal{N}(0, \mathbf{Q}_k), \\ \mathbf{v}_k &\sim \mathcal{N}(0, \mathbf{R}_k). \end{aligned}$$

Hence, $\mathbf{x}_k$ as well as $\mathbf{z}_k$ are both not precise, degraded by noises $\mathbf{w}_k, \mathbf{v}_k$ and the likelihoods for a single measurement and state estimation are

$$p(\mathbf{z}_k|\mathbf{x}_k) = \mathcal{N}(\mathbf{z}_k; \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k), \tag{2.3}$$

$$p(\mathbf{x}_k|\mathbf{x}_{k-1}) = \mathcal{N}(\mathbf{x}_k; F_{k|k-1}\mathbf{x}_{k-1}, \mathbf{Q}_k). \tag{2.4}$$

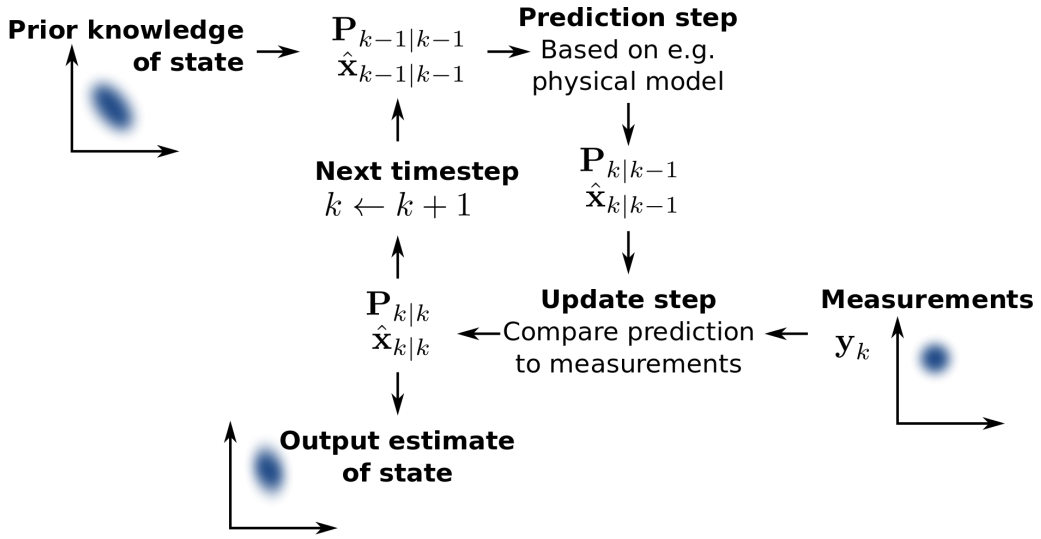


Figure 2.1: The whole process of Kalman filter. We start with an information from the previous step $(k-1|k-1)$, then perform prediction to get expected values $(k|k-1)$, after which we incorporate measurements information and perform an update to get the corrected state $(k|k)$.
Source: https://www.wikiwand.com/en/Kalman_filter

2.3.2 Filter definition

The Kalman Filter is a so-called recursive estimator, needing only state estimates from the previous step to proceed with the next step. The state of the estimator is described by two variables

- $\hat{\mathbf{x}}_{k|k}$ – the a posteriori state estimate at time k,
- $\mathbf{P}_{k|k}$ – the a posteriori error covariance matrix at time k.

The filter can be then described as two consecutive steps: state prediction based on the model and correction based on the observation. The prediction step is defined as

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1|k-1}, \tag{2.5}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}. \tag{2.6}$$

The correction based on the observation is then performed according to

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k + \mathbf{R}_k)^{-1}, \tag{2.7}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}), \tag{2.8}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k|k-1}. \tag{2.9}$$

The whole flow of the filter is nicely visualized in Figure 2.1 on page 18 .

2.4 Distributed Kalman Filter

The distributed extension of the Kalman filter is rather a family of methods than some specific algorithm. Basically, in its generality, instead of having only a single agent performing updates

(2.1)-(2.9), more of these agents – nodes – are independently carrying these out. These nodes are usually connected to some predefined topology, enabling different kinds of communication and hence the algorithm's variants. For example, the nodes may only be able to send their estimates to some master node without getting any feedback (a scenario of a directed graph topology), but a more complex topology may be chosen, such as that the nodes can communicate with each other.

Because of its generality, a formal definition of a Distributed Kalman Filter is not very informative, simply stating that there is some function performed in an arbitrary step with an arbitrary communication given the topology. These are the parameters differentiating between DKF variants more studied in Section 2.7 and defines the complexity of the given approach.

2.5 Optimality

The classic Kalman filter described in Section 2.3 is the optimal linear filter in the mean squared error if and only if all the following conditions are fulfilled [28]:

1. the model is exactly reflecting the reality,
2. the noise is Gaussian white,
3. the covariances of the noise are exactly known and the noise is static.

In reality, this is rarely the case and the consequences of not satisfying these prerequisites must not be overlooked. These are not just some theoretical perfection claims – not taking these into account when applying the filter much often than not leads to rendering the filter completely useless.

Any of these requirements can be unsatisfied in a given application and different methods have been proposed and used to deal with the problem, such as extensions being able to work with different than Gaussian white noise. This work is no exception and tries to deal with the third requirement – the case when the noise covariances are unknown and not static. The proposed algorithm relax this requirement to some degree, extending the possible use cases for the, otherwise exceptionally effective, filter. It is important to emphasize that the scenario when the proposed algorithm is used still needs to satisfy the rest of the requirements – the model must reflect the reality and the noise must be Gaussian white.

2.6 Noise analysis

This subsection extends the previous one explaining why it is so crucial to correctly estimate covariance matrices and what research has been done to combat the problem when it is not feasible.

As can be seen in (2.7)-(2.9), the noise covariance matrices $\mathbf{R}_k, \mathbf{Q}_k$ are not just some negligible factor of the update but play a major role in the estimation. Kalman filter depends on this a priori knowledge to a great extent and a wrong choice leads to significantly degraded performance or even divergence [29]. It is not uncommon though, that it is almost impossible to estimate the covariance in advance or that the noise covariance is changing based on some unknown model. For instance, real-life applications suffering from this fact are navigational systems and there has been an undergoing research in this area [30, 31]. A so-called *adaptive* Kalman filter has been proposed to solve this issue and plethora of its variants have been carried out. An algorithm based on variational Bayes method proposed in [32], which we use in our diffused variant as an underlying KF computation, seems to be a great candidate to solve difficulties one

faces when trying to apply KF in noisy environments and it is the reason why it became our choice.

2.7 Methods

There are bibliographic reviews covering dozens of methods such as an excellent one [23] and the author sees a very little added value of restating everything. Nevertheless, to give a reader at least a glimpse of what has been researched in this area to get the context, a brief discussion of selected methods follows. The selection is carefully crafted in a way that it relates to the problem under study and the selected approach of this thesis.

Let start with Adaptive Kalman Filter, nicely summarized in [32]. A four distinguished categories of methods exist: correlation, maximum likelihood, covariance matching, and Bayesian methods. The Sage–Husa AKF is a covariance matching method, estimating the noise on the maximum a posterior criterion [33][34]. Convergence is not guaranteed for this algorithm though. A different, maximum likelihood method Innovation-based AKF leverage the fact that the innovation sequence of the KF is a white process. The disadvantage is that it needs large amounts of data to work reliably, rendering it inapplicable when this is not plausible [35].

An interesting bridge between Adaptive Kalman Filter and its distributive extension is an approximation of a Bayesian method, interactive multiple model AKF [36]. In this model, multiple nodes operate simultaneously and tries to estimate noise statistic. To some degree, this is close to what this work tries to achieve but without Variational Bayes algorithm. Additionally, the main concern of the algorithm is focused on noise statistics estimation, rather than on the state estimation accuracy as is in this work. From the perspective of diffusion-based strategies, the most relevant piece of work and a great inspiration can be found in [37], where different diffusion strategies are carried out in DKF. An outstanding summary of research in diffusion-based algorithm in the world of DKF can be found in [23] in Table 5 on page 13.

Chapter 3

Distributed Variational Bayes Adaptive Kalman Filtration

This chapter is the core contribution of author's own research. The novelty of this work is twofold:

1. The existing VBAKF [31] is reformulated in terms of estimation with the exponential family forms and conjugate prior distributions defined in [13] and [32]. This significantly simplifies its derivation and yields variables whose subsequent fusion is straightforward.

2. Fusion of estimates in diffusion networks is described based on [38].

3.1 Problem formulation

As has been described in Section 2.6, our aim is to describe a model in which we will infer not only $\mathbf{x}_k$, but also Process Error Covariance Matrix (PECM) $\mathbf{P}_{k|k-1}$ and Measurement Noise Covariance Matrix (MNCM) $\mathbf{R}_k$ as opposed to traditional Kalman filter where these two are considered static (or changing according to some deterministic model).

Based on Section 2.3, the Probabilistic Density Function (PDF) of predictions for $\mathbf{x}_k$ and the likelihood follows Gaussian distribution:

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}, \mathbf{P}_{k|k-1}) = \mathrm{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}), \tag{3.1}$$

$$p(\mathbf{z}_k|\mathbf{x}_k, \mathbf{R}_k) = \mathrm{N}(\mathbf{z}_k; \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k), \tag{3.2}$$

where

- $\mathrm{N}(\cdot; \mu, \Sigma)$ is Gaussian PDF with corresponding mean μ and covariance matrix Σ,
- $\hat{\mathbf{x}}_{k|k-1}$ is the predicted state vector based on the model as in (2.1),
- $\mathbf{P}_{k|k-1}$ is the PECM based on the model as in (2.9).

We emphasize that $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ are both inaccurate because precise $\mathbf{Q}_k$ from the underlying update equations is unknown. One would not have to use the whole machinery if it was known.

3.1.1 Prior distributions for $\mathbf{P}_{k|k-1}$ and $\mathbf{R}_k$

We are going to leverage the benefits of good choice of priors to make the whole solution tractable as outlined in Section 1.5. To ensure that the posterior belongs to the exponential family, we choose the Inverse-Wishart distribution which is in a Bayesian statistics a common choice as a conjugate prior for the covariance matrix of a multivariate normal distribution with a known mean[39]. That is the case for $\mathbf{P}_{k|k-1}$ and $\mathbf{R}_k$, which are both the covariance matrices of Gaussian PDFs as is in equations (3.1) and (3.2).

Definition 7. Inverse-Wishart distribution is a probability distribution on real-value positive-definite matrices. We say that $\mathbf{X}$ follows the Inverse-Wishart distribution with ν degrees of freedom and the scale matrix Ψ, i.e. $\mathbf{X} \sim \mathrm{IW}(\Psi, \nu)$, if $\mathbf{X}^{-1}$ has a Wishart distribution $\mathbf{X} \sim \mathrm{W}(\Psi^{-1}, \nu)$. The density is

$$\frac{|\boldsymbol{\Psi}|^{\frac{\nu}{2}}}{2^{\frac{\nu p}{2}}\Gamma_p(\frac{\nu}{2})}\,|\mathbf{X}|^{-\frac{\nu+p+1}{2}}\,e^{-\frac{1}{2}\mathrm{tr}(\boldsymbol{\Psi}\mathbf{X}^{-1})}, \tag{3.3}$$

where $\boldsymbol{\Psi} > \mathbf{0} \in \mathbb{R}^{p\times p}$ is a positive-definite scale matrix, $\nu > p-1, \nu \in \mathbb{R}$ are degrees of freedom, $\Gamma_p(\cdot)$ is the gamma function and tr is the trace function. The exponential family form of PDF of Inverse-Wishart distributed matrix $\mathbf{X}$ is

$$\exp\Bigg\{\begin{bmatrix}\mathbf{X}^{-1}\\ \ln|\mathbf{X}|\end{bmatrix}^{\!\top}\begin{bmatrix}-\frac{1}{2}\boldsymbol{\Psi} & -\frac{\nu+p+1}{2}\end{bmatrix} - \Big(\frac{\nu}{2}(p\ln 2 - \ln|\boldsymbol{\Psi}|) + \ln\Gamma_p\Big(\frac{\nu}{2}\Big)\Big)\Bigg\}, \tag{3.4}$$

and the corresponding factors of the form from Section 1.5 are

$$\begin{aligned}\eta(\mathbf{X}) =& \begin{bmatrix}\mathbf{X}^{-1}\\ \ln|\mathbf{X}|\end{bmatrix},\\ \Xi(\Psi,\nu,p) =& \begin{bmatrix}-\frac{1}{2}\boldsymbol{\Psi}, & -\frac{\nu+p+1}{2}\end{bmatrix},\\ A(\eta) =& \frac{\nu}{2}(p\ln 2 - \ln|\boldsymbol{\Psi}|) - \ln\Gamma_p\Big(\frac{\nu}{2}\Big).\end{aligned}$$

The first element Ξ_1 of a *pseudovector* Ξ is a matrix, while the second Ξ_2 is a scalar. The advantages of this notation will be clearer when an addition to natural parameters is performed. The expected value of the IW distributed variable is

$$\begin{aligned}\mathrm{E}[\mathbf{X}] =& \frac{\Psi}{\nu-p-1} &(3.5)\\ =& \frac{2}{2\Xi_2+p+1}\Xi_1. &(3.6)\end{aligned}$$

Following from the above, the probabilities for $\mathbf{P}_{k|k-1}$ and $\mathbf{R}_k$ are

$$\begin{aligned}p(\mathbf{P}_{k|k-1}|\mathbf{z}_{1:k-1}) &= \mathrm{IW}(\mathbf{P}_{k|k-1}; \hat{t}_{k|k-1}, \hat{\mathbf{T}}_{k|k-1}), &(3.7)\\ p(\mathbf{R}_k|\mathbf{z}_{1:k-1}) &= \mathrm{IW}(\mathbf{R}_k; \hat{u}_{k|k-1}, \hat{\mathbf{U}}_{k|k-1}). &(3.8)\end{aligned}$$

Prior parameters (hyperparameters) $\hat{t}_{k|k-1}, \hat{\mathbf{T}}_{k|k-1}, \hat{u}_{k|k-1}, \hat{\mathbf{U}}_{k|k-1}$ need to represent the initial knowledge as precisely as possible. These parameters will be later incorporated into prior's hyperparameters in form of Ξ.

Hyperparameters for PNCM

The approach in [32] proposes the following initial parameter setting for the prior information for $\mathbf{P}_{k|k-1}$

$$\frac{\hat{\mathbf{T}}_{k|k-1}}{\hat{t}_{k|k-1}-n-1} = \tilde{\mathbf{P}}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^T + \tilde{\mathbf{Q}}_{k-1}, \tag{3.9}$$

$$\hat{\mathbf{T}}_{k|k-1} = \tau\tilde{\mathbf{P}}_{k|k-1}, \tag{3.10}$$

$$\hat{t}_{k|k-1} = n + \tau + 1, \tag{3.11}$$

where $\tilde{\mathbf{P}}_{k|k-1}, \tilde{\mathbf{Q}}_{k-1}$ are the *nominal* PECM and PNCM, respectively. The latter, together with $\tau \geq 0$, are model parameters and the choice is discussed later. Substituting $\mathbf{P}_{k|k-1}, \hat{t}_{k|k-1}, \hat{\mathbf{T}}_{k|k-1}$ into (3.4) and combining with (3.10) and (3.11), we get

$$\eta(\mathbf{P}_{k|k-1}) = \begin{bmatrix} \mathbf{P}_{k|k-1}^{-1} \\ \ln|\mathbf{P}_{k|k-1}| \end{bmatrix}, \tag{3.12}$$

$$\Xi(\hat{\mathbf{T}}_{k|k-1}, \hat{t}_{k|k-1}, n) = \begin{bmatrix} -\frac{1}{2}\hat{\mathbf{T}}_{k|k-1}, & -\dfrac{\hat{t}_{k|k-1}+n+1}{2} \end{bmatrix} \tag{3.13}$$

$$= \begin{bmatrix} -\frac{1}{2}\tau\tilde{\mathbf{P}}_{k|k-1}, & -\frac{1}{2}\tau - 1 - n \end{bmatrix}. \tag{3.14}$$

where n is the dimension of $\hat{\mathbf{T}}_{\cdot|\cdot}$.

Hyperparameters for MNCM

According to (1.1), the prior for $p(\mathbf{R}_k|\mathbf{z}_{1:k-1})$ is

$$p(\mathbf{R}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{R}_k|\mathbf{R}_{k-1})p(\mathbf{R}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{R}_{k-1}. \tag{3.15}$$

To make sure that $p(\mathbf{R}_k|\mathbf{z}_{1:k-1})$ is also Inverse-Wishart distributed, the $p(\mathbf{R}_k|\mathbf{R}_{k-1})$ must have a suitable form so the Bayesian update results in the exponential family. Allowing slowly varying MNCM, as is the case in many real-world applications, a simple heuristic from [40] is chosen to deal with this uncertainty. That is spreading the previous posterior by a factor of $\rho > 0.95$, which is then an additional model parameter

$$\hat{u}_{k|k-1} = \rho(\hat{u}_{k-1|k-1} - m - 1) + m + 1, \tag{3.16}$$

$$\hat{\mathbf{U}}_{k|k-1} = \rho\hat{\mathbf{U}}_{k-1|k-1}, \tag{3.17}$$

where m is the size of matrix $\hat{\mathbf{U}}_{\cdot|\cdot}$. The initial MNCM $\mathbf{R}_0$ is also considered to be Inverse-Wishart distributed

$$p(\mathbf{R}_0) = \mathrm{IW}(\mathbf{R}_0; \hat{u}_{0|0}, \hat{\mathbf{U}}_{0|0}). \tag{3.18}$$

Similarly to the PNCM case in (3.9), the initial prior information for the mean value of $\mathbf{R}_0$ is set as the initial nominal MNCM $\tilde{\mathbf{R}}_0$

$$\frac{\hat{\mathbf{U}}_{0|0}}{\hat{u}_{0|0} - m - 1} = \tilde{\mathbf{R}}_0, \tag{3.19}$$

$\tilde{\mathbf{R}}_0$ then becoming as an additional model parameter.

Substituting $\mathbf{R}_{k-1}, \hat{u}_k, \hat{\mathbf{U}}_k$ into (3.4) and combining with (3.17) and (3.16), we get

$$\eta(\mathbf{R}_{k-1}) = \begin{bmatrix} \mathbf{R}_{k-1}^{-1} \\ \ln|\mathbf{R}_{k-1}| \end{bmatrix}, \tag{3.20}$$

$$\Omega(\hat{\mathbf{U}}_k, \hat{u}_k, m) = \begin{bmatrix} -\frac{1}{2}\hat{\mathbf{U}}_k, & -\dfrac{\hat{u}_k + m + 1}{2} \end{bmatrix} \tag{3.21}$$

$$= \begin{bmatrix} -\frac{1}{2}\rho\hat{\mathbf{U}}_k, & -\frac{1}{2}\rho(\hat{u}_k - m - 1) - m - 1 \end{bmatrix}, \tag{3.22}$$

where n is the dimension of $\hat{\mathbf{U}}_k$.

Choice of the nominal PNCM and MNCM

In can be found in the detailed analysis [32] that to guarantee estimator convergence, the initial nominal PNCM $\tilde{\mathbf{Q}}_k$ needs to be near the initial true PNCM $\mathbf{Q}_k$. In the paper and as well in this work, we set $\mathbf{Q}_k = \text{diag}[\alpha_{1,k}, \ldots, \alpha_{n,k}], \alpha_{i,k} > 0$. In reality, this is done based on the state-of-the-art engineering knowledge in the domain. The same is applied for the initial nominal MNCM $\tilde{\mathbf{R}}_0$ and the choice is $\mathbf{R}_k = \text{diag}[\beta_1, \ldots, \beta_m], \beta_i > 0$. The reader is kindly redirected to the original paper for more information.

3.2 Variational Bayes derivation

In order to infer all $\mathbf{x}_k, \mathbf{P}_{k|k-1}$ and $\mathbf{R}_k$, a posterior PDF $p(\mathbf{x}_k, \mathbf{P}_{k|k-1}, \mathbf{R}_k)$ needs to be estimated. As thoroughly described in Section 1.6, the Variational Bayes expects the following approximation

$$p(\mathbf{x}_k, \mathbf{P}_{k|k-1}, \mathbf{R}_k|\mathbf{z}_{1:k}) \approx q(\mathbf{x}_k)q(\mathbf{P}_{k|k-1})q(\mathbf{R}_k), \tag{3.23}$$

where $q(\mathbf{x}_k), q(\mathbf{P}_{k|k-1}), q(\mathbf{R}_k)$ are factors of our factorized distribution $q(\cdot)$ by which we approximate the $p(\cdot)$. These factors are obtained using Variational Bayes machinery including minimizing the Kullback-Leibler divergence and using CAVI.

3.2.1 Probabilistic model

Combining (3.1), (3.2), (3.7) and (3.8), the joint PDF can be factored as

$$p(\boldsymbol{\Xi}, \mathbf{z}_{1:k}) = p\big(\mathbf{z}_k|\mathbf{x}_k, \mathbf{R}_k\big)p\big(\mathbf{x}_k|\mathbf{z}_{1:k-1}, \mathbf{P}_{k|k-1}\big)p(\mathbf{P}_{k|k-1}|\mathbf{z}_{1:k-1})p(\mathbf{R}_k|\mathbf{z}_{1:k-1})p(\mathbf{z}_{1:k-1}), \tag{3.24}$$

and substituting appropriate distribution for these PDFs:

$$\begin{aligned} p\big(\boldsymbol{\Xi}, \mathbf{z}_{1:k}\big) = {} & \mathrm{N}\big(\mathbf{z}_k; \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k\big)\mathrm{N}\big(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}\big) \\ & \times \mathrm{IW}\big(\mathbf{P}_{k|k-1}; \hat{t}_{k|k-1}, \hat{\mathbf{T}}_{k|k-1}\big)\mathrm{IW}\big(\mathbf{R}_k; \hat{u}_{k|k-1}, \hat{\mathbf{U}}_{k|k-1}\big) \\ & \times p\big(\mathbf{z}_{1:k-1}\big). \end{aligned}$$

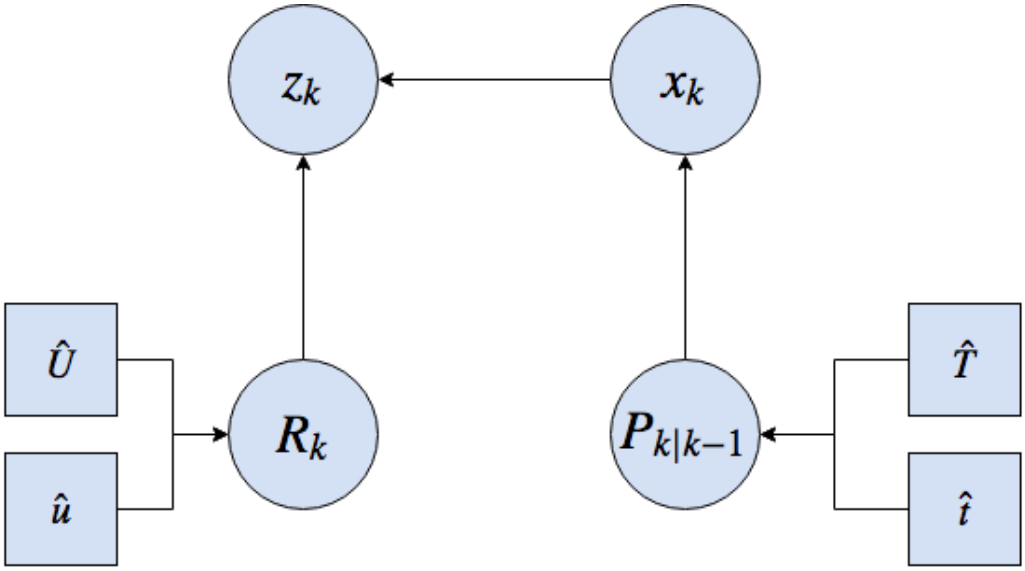


Figure 3.1: Graphical model of probabilistic model (3.24)

3.2.2 Deriving update equations

As opposed to [32], we are going to leverage exponential family form to infer the update equations. Considering the exponential form of a Gaussian distribution

$$\mathrm{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \exp\Big\{ \begin{bmatrix} \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\ -\frac{1}{2}\boldsymbol{\Sigma}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}, & \mathbf{x}\mathbf{x}^T \end{bmatrix} - (\frac{1}{2}\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \frac{1}{2}\ln|\boldsymbol{\Sigma}|) \Big\} \tag{3.25}$$

$$= \exp\Big\{ \begin{bmatrix} \boldsymbol{\Sigma}^{-1} \\ \ln|\boldsymbol{\Sigma}^{-1}| \end{bmatrix} \begin{bmatrix} -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T, & -\frac{1}{2} \end{bmatrix} + \ln(2\pi) \Big\} \tag{3.26}$$

where $\mathbf{x}\mathbf{x}^{\mathbf{T}}$ is an outer product, hence resulting into a matrix of $\mathbb{R}^{n\times n}$. Substituting appropriate factors for both $\mathbf{x}_k$ as well as $\mathbf{z}_k$ we get

$$\mathrm{N}(\mathbf{x}_k;\hat{\mathbf{x}}_{k|k-1},\mathbf{P}_{k|k-1}) = \exp\Big\{ \begin{bmatrix} \mathbf{P}^{-1}_{k|k-1} \\ \ln|\mathbf{P}^{-1}_{k|k-1}| \end{bmatrix} \begin{bmatrix} -\frac{1}{2}(\mathbf{x}_k-\hat{\mathbf{x}}_{k|k-1})(\mathbf{x}_k-\hat{\mathbf{x}}_{k|k-1})^T, & -\frac{1}{2} \end{bmatrix} - k_1 \Big\}, \tag{3.27}$$

$$\mathrm{N}(\mathbf{z}_k;\mathbf{x}_k,\mathbf{R}_k) = \exp\Big\{ \begin{bmatrix} \mathbf{R}^{-1}_{k} \\ \ln|\mathbf{R}^{-1}_{k}| \end{bmatrix} \begin{bmatrix} -\frac{1}{2}(\mathbf{z}_k-\mathbf{x}_k)(\mathbf{z}_k-\mathbf{x}_k)^T, & -\frac{1}{2} \end{bmatrix} - k_2 \Big\}. \tag{3.28}$$

As can be seen, the natural parameters match for (3.27) and (3.12) as well as for (3.28) and (3.20), which is of course intentional and result of a clever choice of the conjugate priors.

With respect to (1.5), the single step Bayesian update of Inverse-Wishart hyperparameters with Multivariate Normal Distribution can be written as

$$\Xi_i \leftarrow \underbrace{\begin{bmatrix} -\frac{1}{2}\boldsymbol{\Psi} \\ -\dfrac{\nu+p+1}{2} \end{bmatrix}}_{\Xi_{i-1}} + \underbrace{\begin{bmatrix} -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T \\ -\frac{1}{2} \end{bmatrix}}_{T(x,\mu)}. \tag{3.29}$$

Utilizing the fact that (as in [32])

$$\mathrm{E}^i[-\frac{1}{2}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})^T] = \mathbf{P}^i_{k|k} + (\hat{\mathbf{x}}^i_{k|k} - \hat{\mathbf{x}}_{k|k-1})(\hat{\mathbf{x}}^i_{k|k} - \hat{\mathbf{x}}_{k|k-1})^T, \tag{3.30}$$

$$\mathrm{E}^i[-\frac{1}{2}(\mathbf{z}_k - \mathbf{H}_k\mathbf{x}_k)(\mathbf{z}_k - \mathbf{H}_k\mathbf{x}_k)^T] = \mathbf{H}_k\mathbf{P}^i_{k|k}\mathbf{H}_k^T + (\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}^i_{k|k})(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}^i_{k|k})^T, \tag{3.31}$$

the single step i of the variational updates are

$$\Xi^n_{i+1} \leftarrow \Xi^{n-1} + \begin{bmatrix} -\frac{1}{2}(\mathbf{P}^i_{k|k} + (\hat{\mathbf{x}}^i_{k|k} - \hat{\mathbf{x}}_{k|k-1})(\hat{\mathbf{x}}^i_{k|k} - \hat{\mathbf{x}}_{k|k-1})^T) \\ -\frac{1}{2} \end{bmatrix}, \tag{3.32}$$

$$\Omega^n_{i+1} \leftarrow \Omega^{n-1} + \begin{bmatrix} \mathbf{H}_k\mathbf{P}^i_{k|k}\mathbf{H}_k^T + (\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}^i_{k|k})(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}^i_{k|k})^T \\ -\frac{1}{2} \end{bmatrix}, \tag{3.33}$$

where Ξ^{n-1}, Ω^{n-1} are hyperparameters obtained from the results of Kalman filter of the previous observation, derived using formulas (3.14) and (3.22).

Kalman correction

The last bit of the single CAVI iteration is the Kalman correction, which is an iterative extension of (2.7)-(2.9)

$$\mathbf{K}_k^{(i+1)} = \hat{\mathbf{P}}^{(i+1)}_{k|k-1}\mathbf{H}_k^T\big(\mathbf{H}_k\hat{\mathbf{P}}^{(i+1)}_{k|k-1}\mathbf{H}_k^T + \hat{\mathbf{R}}_k^{(i+1)}\big)^{-1}, \tag{3.34}$$

$$\hat{\mathbf{x}}^{(i+1)}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k^{(i+1)}\big(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}\big), \tag{3.35}$$

$$\mathbf{P}^{(i+1)}_{k|k} = \hat{\mathbf{P}}^{(i+1)}_{k|k-1} - \mathbf{K}_k^{(i+1)}\mathbf{H}_k\hat{\mathbf{P}}^{(i+1)}_{k|k-1}. \tag{3.36}$$

3.3 Distribution and parameter diffusion

Thanks to the usage of exponential family form, parameter diffusion can be easily incorporated into the final algorithm. The nodes only need to exchange the hyperparameters Ξ^n and Ω^n. While the previous chapters were mostly based on the algorithm described in [32], the parameter diffusion part is based on [13]. The result algorithm is presented here, but the reader is advised to reach to the latter paper for more details on the diffusion.

Consider a network of connected nodes, each of them obtaining their own observation. A single node i obtain a measurement in time t and proceed with Kalman filtration described as above. The node then *posses* $\mathbf{P}^{i,t}_{k|k}, \hat{\mathbf{x}}^{i,t}_{k|k}$ and also result hyperparameters $\Xi^{i,t}, \Omega^{i,t}$. It then *contacts* its neighbor nodes $1, ..., d$ and attempts to gather all available hyperparameters $\Xi^{j,t}, \Omega^{j,t}, j \in \{1, ..., d\}$ based on the current conditions (for instance, only a single hop distance may be available). The node then performs a parameter fusion of these hyperparameters based on a chosen strategy represented by some function $g(\Xi^{i,t}, \Xi^{1,t}, ..., \Xi^{d,t})$ to obtain a new merged $\Xi^{i,t}_*$. An example of such function may be just a simple averaging over all hyperparameters $\Xi^{i,t}_* = g(\Xi^{i,t}, \Xi^{1,t}, ..., \Xi^{d,t}) = \frac{\sum_{s=\{i,1,...,d\}} \Xi^{s,t}}{d+1}$. The situation for the hyperparameter $\Omega^{i,t}$ is analogical. The node then sends this updated hyperparameters $\Xi^{i,t}_*, \Omega^{i,t}_*$ back to the cooperating nodes, which incorporate this hyperparameter into their filter state.

3.4 Algorithm summary

Here we summarize the previous pages into a single algorithm. A single step of the proposed algorithm for a single node is then

Algorithm 8. *1.* ***Inputs****:* $\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1}, \hat{u}_{k-1|k-1}, \hat{\mathbf{U}}_{k-1|k-1}, \mathbf{F}_{k-1}, \mathbf{H}_k, \mathbf{z}_k, \tilde{\mathbf{Q}}_{k-1}, m, n, \tau, \rho, N$

2. Time Update of the Kalman Filter as in (2.5) *and* (2.6)

3. Initialize Ξ_{init} *by* (3.14), Ω_{init} *by* (3.22), $\hat{\mathbf{x}}_{k|k}^{(0)} = \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k}^{(0)} = \tilde{\mathbf{P}}_k$

For *i in* N ***do:***

4. Get new Ω^i *based on the* (3.33) *and compute* $\mathrm{E}^i[\mathbf{R}_k]$ *using* (3.6)

5. Get new Ξ^i *based on the* (3.32) *and compute* $\mathrm{E}^i[\mathbf{P}_{k|k-1}]$ *using* (3.6)

6. Perform Kalman correction (3.34)-(3.36) *using expected values to obtain corrected* $\hat{\mathbf{x}}_{k|k}^{i+1}, \mathbf{P}_{k|k-1}^{i+1}$

end for

7. Perform parameter diffusion, i.e. if hyperparameters from nodes $j \in \{1, ..., d\}$ *are available, get*

$$\Xi_*^{i,t} = g(\Xi^{i,t}, \Xi^{1,t}, ..., \Xi^{d,t}) \tag{3.37}$$

$$\Omega_*^{i,t} = g(\Omega^{i,t}, \Omega^{1,t}, ..., \Omega^{d,t}) \tag{3.38}$$

8. ***Returns****:* $\hat{\mathbf{x}}_{k|k}^{i+1}, \mathbf{P}_{k|k-1}^{i+1}, \Xi_*^{i,t}, \Omega_*^{i,t}$

Chapter 4

Results and simulations

The following chapter offers an in-depth analysis of the performance of the proposed algorithm. To guarantee a reproducible research, implementation details are provided and the code with the comparison is publicly available on `https://github.com/hnykda/kfsims`. This thesis focused mostly on state prediction accuracy and less on the noise analysis, although even the latter is mentioned when it helps to understand mechanisms explaining the behavior.

Please note that some figures should be compared to each other, rather than being useful on its own. Sometimes, it is the trend of all plotted lines what is interesting rather then individual lines. This is true especially in the case of the multi-node variants when the author's goal is to demonstrate that all the nodes are having some specific property which would be hard to demonstrate by plotting only a single line.

4.1 Implementation

The implementation used in this work has been written in feature-rich Python language ecosystem. Next to the CPython interpreter 3.6, following libraries (all licensed under MIT license) has been utilized:

- scipy 1.0.0 – generator of pseudo-random numbers from various distributions,
- numpy 1.14.2 – to store arrays and perform mathematical operations, especially linear algebra ones,
- networkx 2.1 – to simulate various network topology types of the measurement nodes,
- jupyter 5.2.3 – "notebook" ecosystem for a rapid analysis and environment to plot the charts,
- filterpy 1.2.1 – the reference implementation of a classic Kalman Filter used in comparison,
- matplotlib 2.2.2 – plotting the charts.

All the code is available on GitHub page `https://github.com/hnykda/kfsims` including the instruction how to replicate the analysis and testing environment. Most of the functionality has been written from scratch by the author while having a reference implementation by the supervisor (plus the model of the testing trajectory). The code is structured into several parts, notably objects `MeasurementNode` and `IWPrior` plus many of utility functions in the appropriate modules such as `exponential_family.py` or `network.py`, which should be self-explanatory.

Comparison

The notation for the components of the state vector is kept as x_i in all the charts. From the nature of the problem under study, it is clear that the first two components which are observable behave in a similar way and the same is true for the latter two (which are not observable). To keep the analysis clean and concise, only a representative component is plotted if the behavior of the two is not significantly different.

The main metric used for the comparison is the root mean square error (RMSE). Where appropriate, standard deviation of errors is listed. The result numbers discussed further are results of many iterations (more than 20) with different initial noise assigned to different nodes. It was although made sure that when comparing two different variants, same noise has been attributed to the appropriate elements. Finally, in situations when we are going to *visualize* matrices behavior a Frobenius matrix norm is used. The matrices are often close to a matrix $c\mathrm{I}, c > 0$, which justifies this simplification.

4.2 Testing data

Here we describe the dynamical system model of a 2D tracking problem in the standard notation as in Section 2.3 suitable for the Kalman filter:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & t & 0 \\ 0 & 1 & 0 & t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{Q} = 2 \cdot \begin{bmatrix} t^3/3 & 0 & t^2/2 & 0 \\ 0 & t^3/3 & 0 & t^2/2 \\ t^2/2 & 0 & t & 0 \\ 0 & t^2/2 & 0 & t \end{bmatrix}, t = 0.1$$

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \mathbf{R} = 0.25 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

A random trajectory was then simulated for all four components using this model with multivariate Gaussian random noise with covariance $\mathbf{Q}$ added to the each "true" data point as well as to the observations with covariance $\mathbf{R}$. A possible result of the first two components of the test trajectory can be seen on Figure 4.1 on page 30 while the other two on Figure 4.2 on page 30. We always generated 250 time steps.

4.2.1 Noise

Different types of noise were tested to thoroughly examine the behavior of the filter under different circumstances. Examples for a static noise and variable noise can be seen in Figure 4.3 on page 31 and Figure 4.4 on page 32. In the simulations, both observable components were distorted by the type of the noise under study. Also, for the sake of simplicity, MNCM was the factor we considered that can be slowly varying. In cases when the true covariance is unknown, the initial covariance was set to a base trajectory covariance $\mathbf{R}$, simulating an estimate one could obtain in the real application by other means (for instance, by historical estimates or from the previous research).

It is worth mentioning that because of the task's triviality, the CAVI algorithm usually converged in few iterations – less than 5. Therefore, the number of iterations is set to the conservative 10. There is an exhaustive analysis of the algorithm stability and parameters effect in [32].

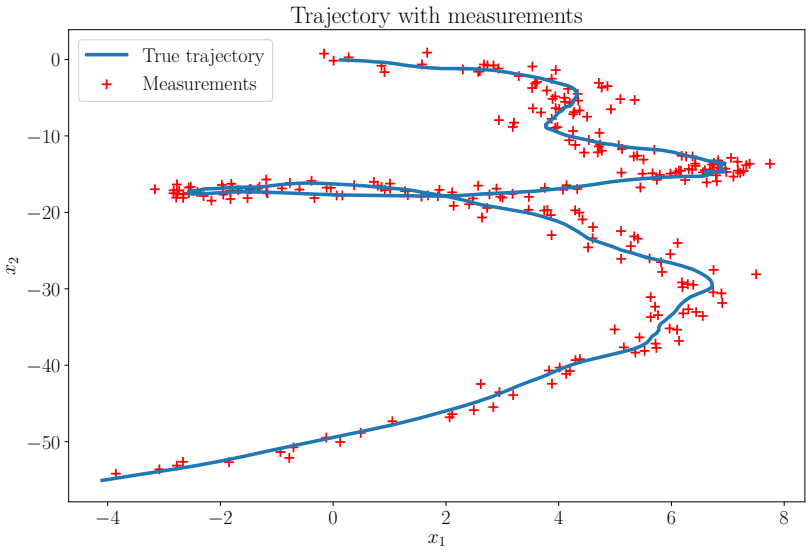


Figure 4.1: The first two components of the state vector plotted against each other. The red crosses mark the noisy measurements of these observable components.

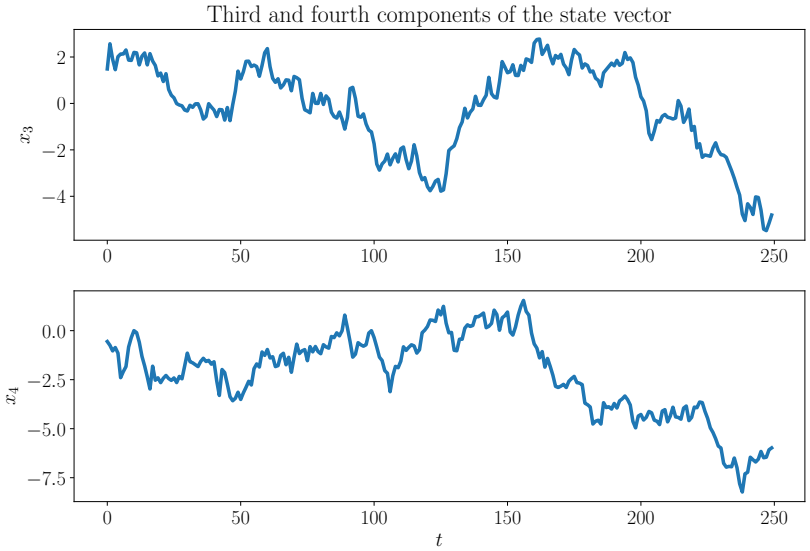


Figure 4.2: True third and fourth component of the state vector. There are no measurements as opposed to the Figure 4.1 on page 30 since these two variables are not observable.

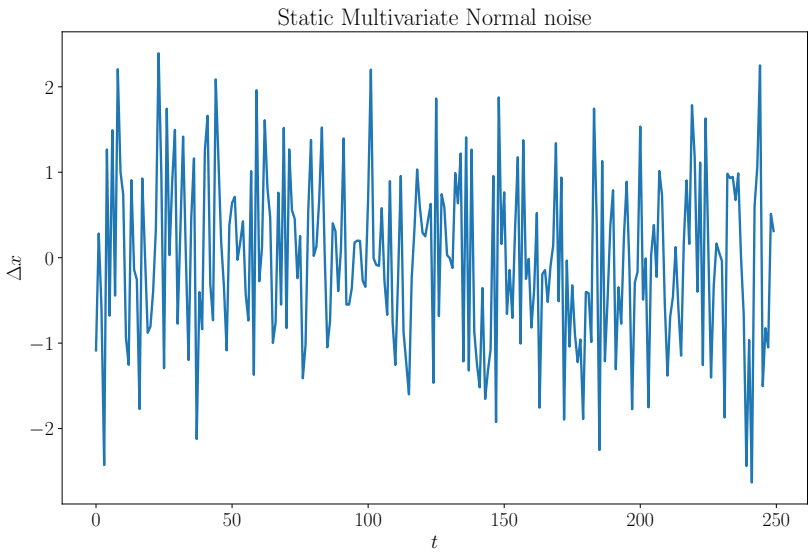


Figure 4.3: Static Multivariate Gaussian Random noise.

4.3 Single node VBAKF

In this section, we compare the VBAKF algorithm performance against the classic Kalman Filter (CKF). It is important to distinguish between the situation where the true noise covariance is known and where it is not. Another property of the system is the variance of the noise covariance over time. Since the motivation of developing VBAKF is especially for the situations when the covariance is unknown or there is a risk of choosing a wrong one, it is clear where the VBAKF is expected to perform better.

When the true covariance is known and static – an unlikely situation in the real world systems – we do not expect any improvements in predictions of VBAKF over CKF. Actually, the performance may be degraded by VBAKF trying to adjust to the noise and because variational Bayes is just an approximation, making the overall accuracy score worse than CKF. In the case when the covariance is known and changing depends on how is one able to tune the algorithm parameters (such as ρ or τ), how big the changes are and how imprecise is the a priori value. Therefore, the main advantage is in the situations when one does not know the true value a priori or it is varying. It is the scenario where we expect VBAKF to surpass CKF, since if we are at least somewhat close to the initial estimate, it should correct itself to the more precise values over time. Additionally, if the covariance is slowly changing, than VBAKF is able to adjust it as well.

4.3.1 Static noise

As it is clear from the Figure 4.5 on page 32, the claim mentioned in the previous section is validated. The CKF variant is slightly outperforming VBAKF, although both variants are performing quite well. Figure 4.6 on page 33 and Figure 4.7 on page 33 allow for some introspection to the observed behavior. VBAKF algorithm seems to pay the price for being more flexible than

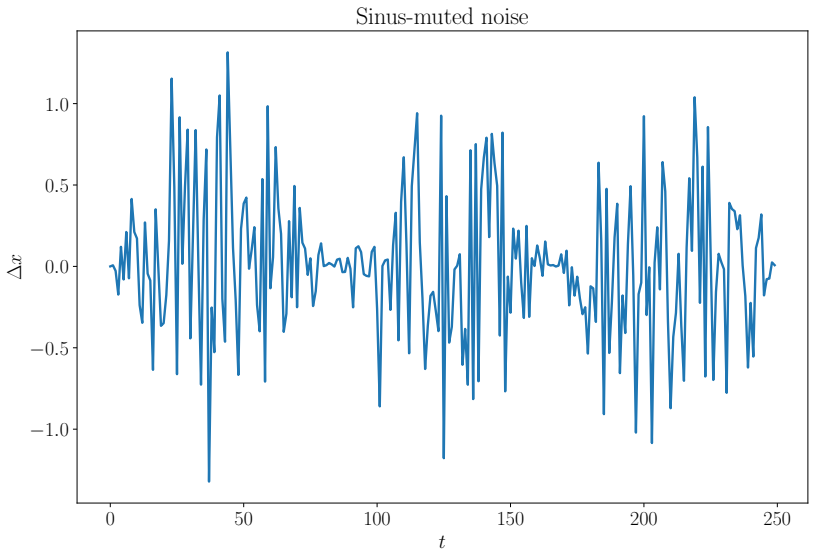


Figure 4.4: A variable variance in the noise is simulated in this example.

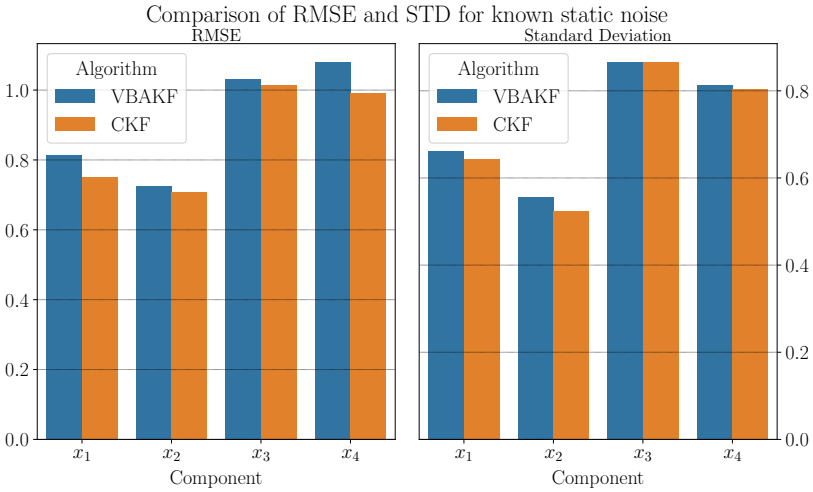


Figure 4.5: As expected, there is no performance gain in prediction(left) nor standard deviation (right) using VBAKF in the simplest possible situation of the known static noise.

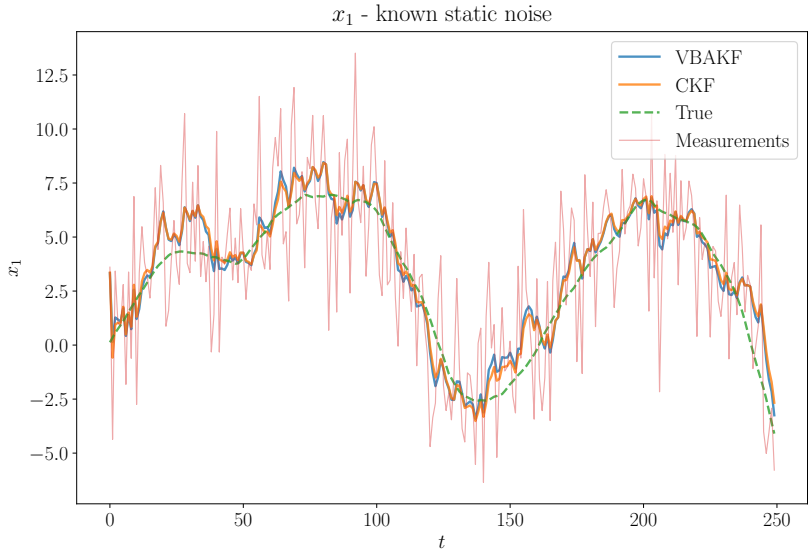


Figure 4.6: The performance of both filters for x_1 together with the measurements and the true trajectory.

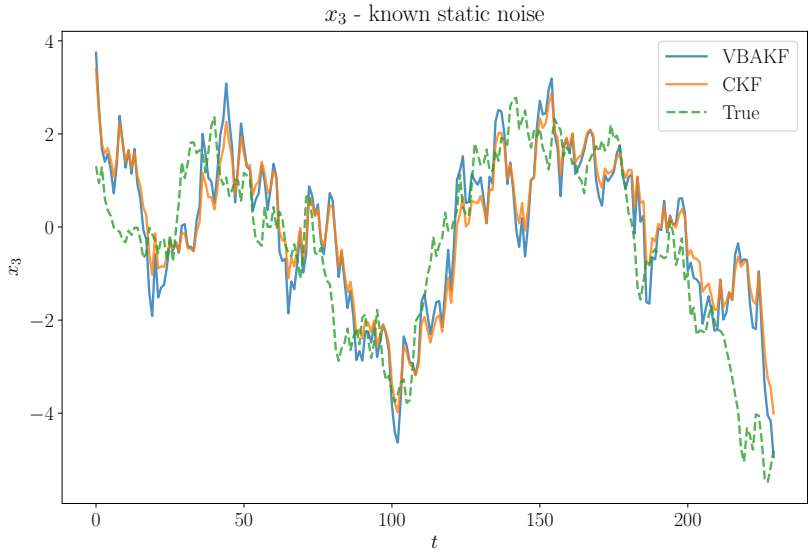


Figure 4.7: The performance of both filters for x_3 together with the true trajectory. This is purely estimated variable and not directly observed, hence the measurements are missing.

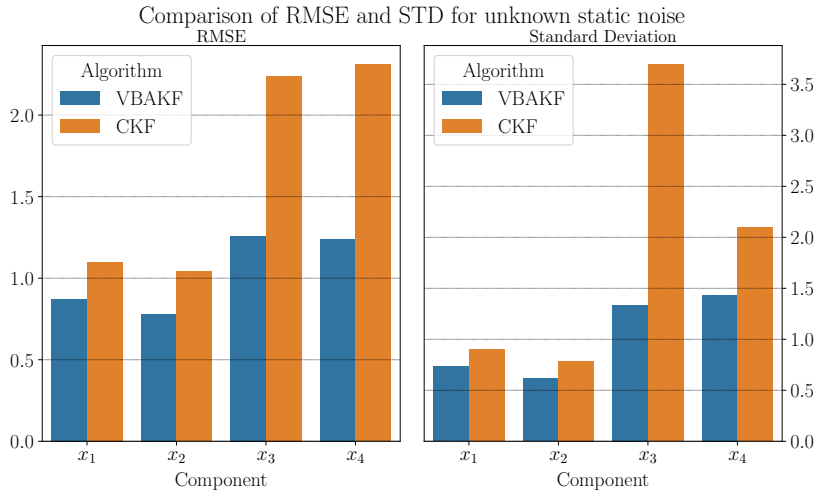


Figure 4.8: Although the performance gain in RMSE of the observed components is already better (˜20%), the major difference is present in the unobserved ones. VBAKF is performing over 2 times better in terms of RMSE (left) and 3 times better in terms of standard deviation of RMSE (right).

CKF and it is following the noise peaks too much. This could be to some degree controlled by the τ hyperparameter, but not much without degrading performance in some other means, such as becoming too rigid, or coalesce with CKF.

Let us also demonstrate the situation when the true covariance is not known and set to some reasonable estimate close to the true covariance. The remarkable property of being able to adjust to the true covariance has been confirmed, as can be seen from RMSE comparison in Figure 4.8 on page 34, the performance is significantly better, especially in the unobserved components of the state vector. This is even more interesting when considering that the standard deviation of prediction error in Figure 4.8 on page 34 is significantly lower for VBAKF than in CKF. This is the remarkable property of the proposed algorithm and figures for the first component in Figure 4.9 on page 35 and third component in Figure 4.10 on page 35 only supports that. Both algorithms started with the noise covariance estimate which was a bit off the true value, but VBAKF variant could adjust for it, placing less emphasis on the measurements and believing more to the prediction step. CKF algorithm just stays on the same level of reliance without correction.

4.3.2 Variable noise

In this scenario, it is again considered that the value for the true covariance is unknown and contrary to the previously discussed static case, it is slowly changing as is visualized on the Figure 4.4 on page 32. From the performance evaluation perspective, the situation is analogous to the previous case when the noise covariance is not known. Based on the expectations from the theory and the previous results, there is no surprise that VBAKF is performing significantly better over CKF with details in Figure 4.11 on page 36. As can be seen from trajectories on Figure 4.12 on page 36 and Figure 4.13 on page 37, VBAKF is again able to put less emphasis on the measurements during epochs of high noise, contrary to CKF being to sensitive in during these.

Finally, a visualization of VBAKF capabilities can be seen on the graph Figure 4.14 on page 38. The adaptation of the noise covariance matrix to the true unknown covariance matrix is

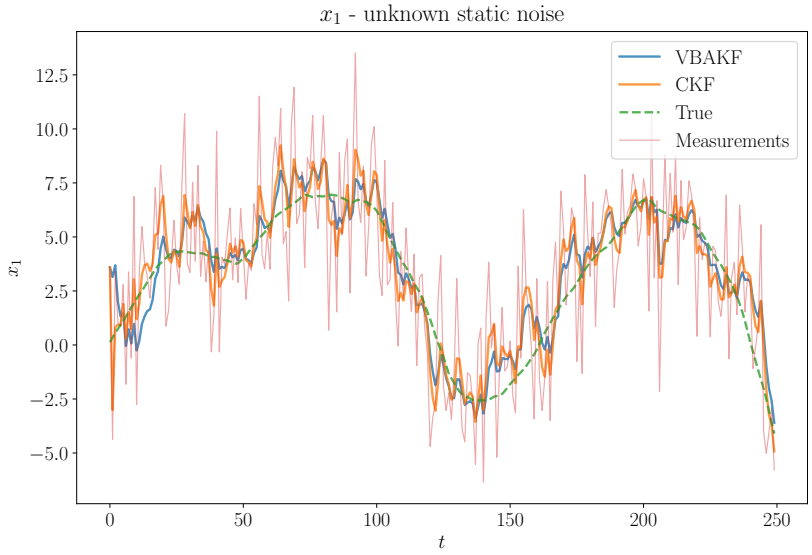


Figure 4.9: The comparison of both filters for the first state component with the measurements and the true trajectory clearly shows that the VBAKF variant can "correct" the imprecise a priori estimation to a more suitable value.

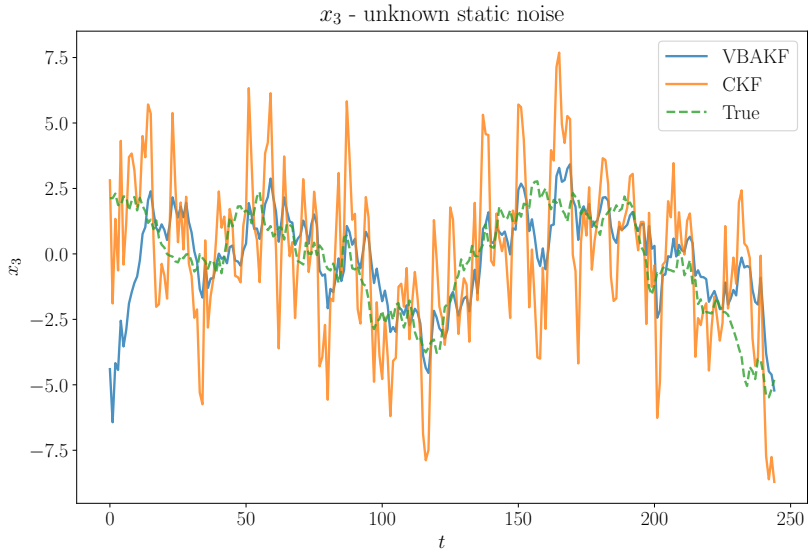


Figure 4.10: Estimated values of x_3 component by both algorithms and with the true trajectory and the measurements. As the initial estimate of the covariance was imprecise, VBAKF adjusted for it while CKF did not. First 5 points has been omitted from this plot for a clarity, where both algorithms have quite high peaks before they "catch up" to the trend.

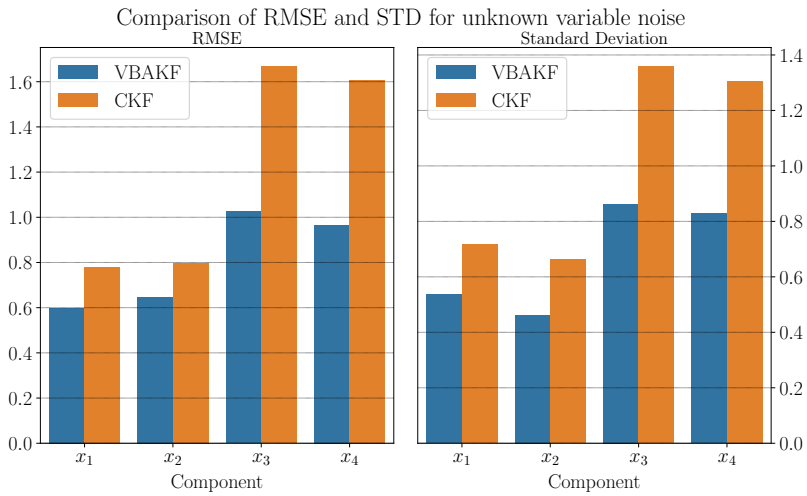


Figure 4.11: Similarly to the unknown static noise case, VBAKF is clearly performing significantly better in terms of RMSE (left) with much less variance in the estimation error (right). Furthermore, this is exhibited over all components.

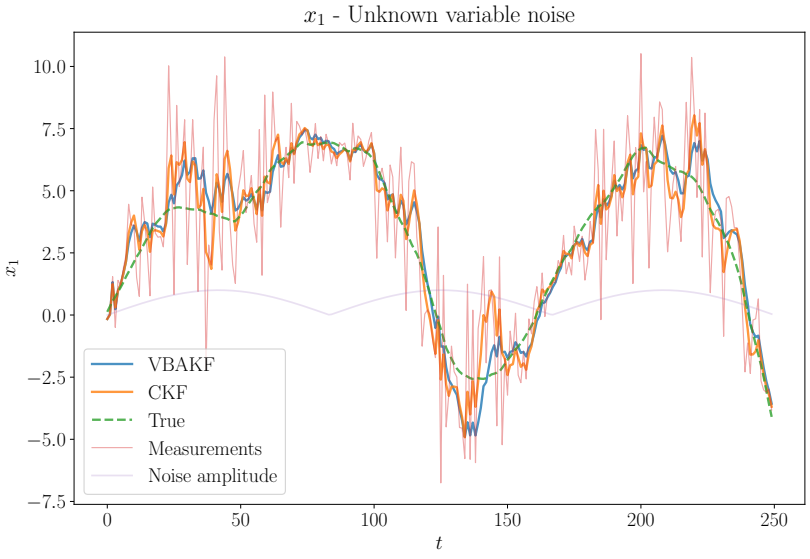


Figure 4.12: First component estimates for both algorithms with observations, true values and also the noise amplitude. Notice how VBAKF is becoming more rigid during high noise peaks while still being flexible to catch up to the trajectory change.

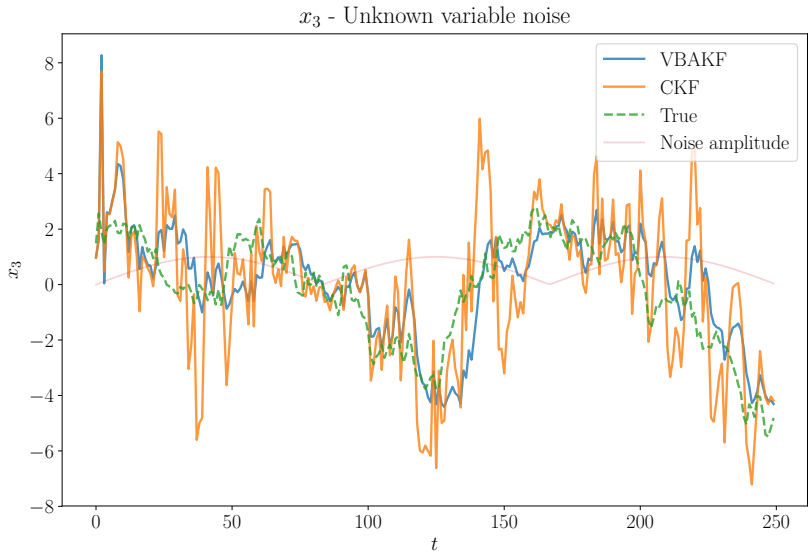


Figure 4.13: Third component estimates for both algorithms with observations, true values and also the noise amplitude. As in the case of Figure 4.12 on page 36, the VBAKF can control during very noisy intervals.

supporting the earlier findings of being able to account for noisy epochs.

4.3.3 Conclusion

The simulations show that our single node VBAKF implementation is behaving as the theory predicts and in concordance with the reference source [32] where the more emphasis on noise analysis was carried out. VBAKF does not perform better in the case of a static noise when the covariance is known, merely just adding more computational time and possibly worse results trying to adjust to the random fluctuations. It outperforms CKF in situations when the covariance of the noise is either unknown or slowly changing. The performance of VBAKF was consistently better than CKF in these cases over many iterations and many variants of a noise. It is also worth mentioning that a very little tuning had to be done to get to these results. How close the initial estimate must be to guarantee divergence is of course almost impossible to say in the real world cases, but in our testing case, even extreme initialization lead to a quick correction of the state (in less than 5 observations). Such an extreme initialization was, for example, over three orders of magnitudes of the reality – a situation in which the CKF just does not provide any useful information.

4.4 Distributed VBAKF

Based on the results from the 4.3, only a slowly changing noise is going to be considered to not clutter the analysis of VBAKF with diffusion versus VBAKF without diffusion. As has been mentioned at the beginning of this whole chapter, the figures when plotting information from multiple nodes should be read as a general trend exhibited by the system. It would be a mistake trying to interpret them line by line.

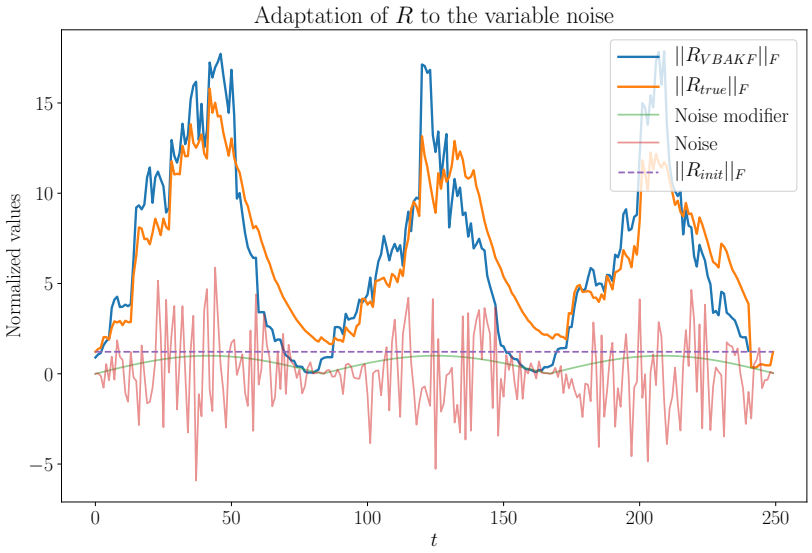


Figure 4.14: Frobenius norms comparison of the VBAKF algorithm and covariance of a moving window (size 20 steps to the future) of the noise. The diagonal values of **R** are increasing during the noise peaks, while going down during low noise epochs. The scale in this chart is not important, it is the trend of the norms. Those should be ideally as close as possible.

4.4.1 Setting

Our network topology of nodes is for simplicity a 4-regular graph as is in Figure 4.15 on page 39. A slowly changing, different noise is attributed to each node in the network (Figure 4.16 on page 39). When we compare diffused versus non-diffused variant, the attributed noise in the both runs was the same in corresponding nodes to make the results comparable and reproducible. The results are averaged over at least 10 iterations.

4.4.2 Estimation performance

As expected, the estimation performance of the diffused variant is better than non-diffused variant. This is not only a result in the averaged results on the figure Figure 4.17 on page 40, but it was consistent across literally all runs. In other words, there was not a *single* run where the non-diffused variant outperforms the diffused one in the matter of Mean RMSE and standard deviation in Figure 4.17 on page 40. The price is the higher run time, since the diffusion variant does an additional step – diffusing parameters – in each iteration. In our setting, this was almost negligible, taking only about 5% longer than the non-diffused variant. This time penalty would, of course, depend on the chosen diffusion strategy.

4.4.3 Noise covariance analysis

The idea behind diffusion optimization is that the *averaging* over several nodes will eliminate extremes and smooths the estimation. This is clearly happening, as can be seen in Figure 4.18 on page 40, where we plot a Frobenius norm of the noise covariance matrix before and after diffusion step. Naturally, this phenomenon was exhibited on all nodes in the network, visualized in Figure 4.19 on page 41.

Another view of this phenomenon is presented in Figure 4.20 on page 41 and Figure 4.21 on

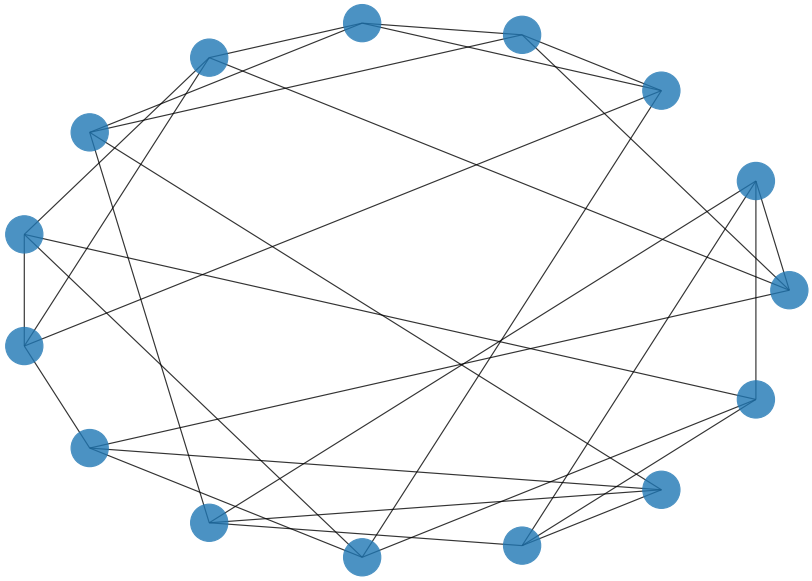


Figure 4.15: A network representing the connection of different measurement nodes connected with the neighbors. The topology is a 4-regular graph.

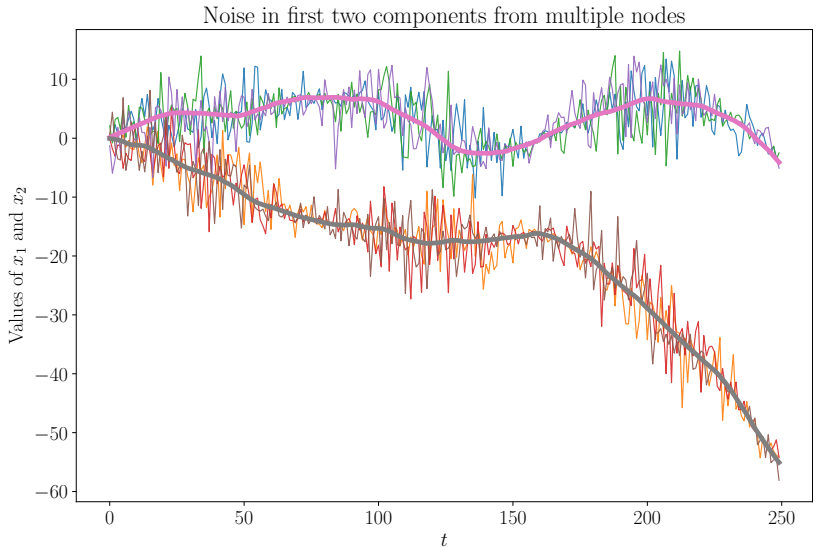


Figure 4.16: Values of the first and second component with noise for different nodes.

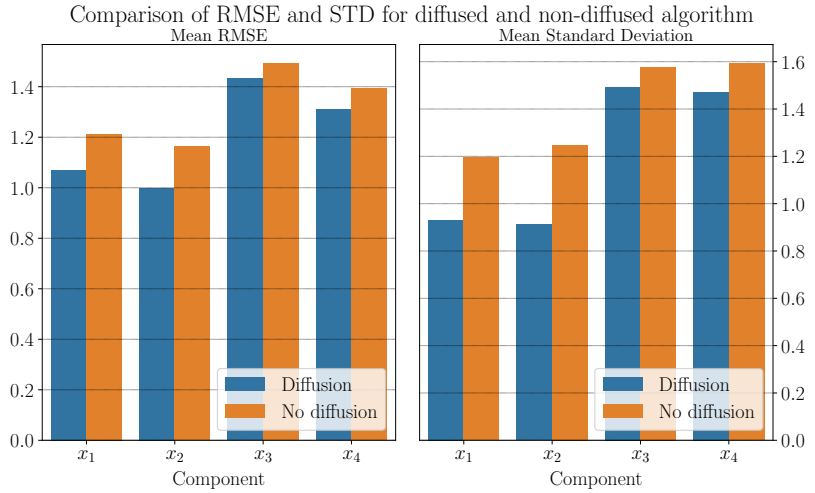


Figure 4.17: Diffusion variant outperforms the non-diffused variant not just in Mean RMSE (left), but also in mean standard deviation over all runs (right).

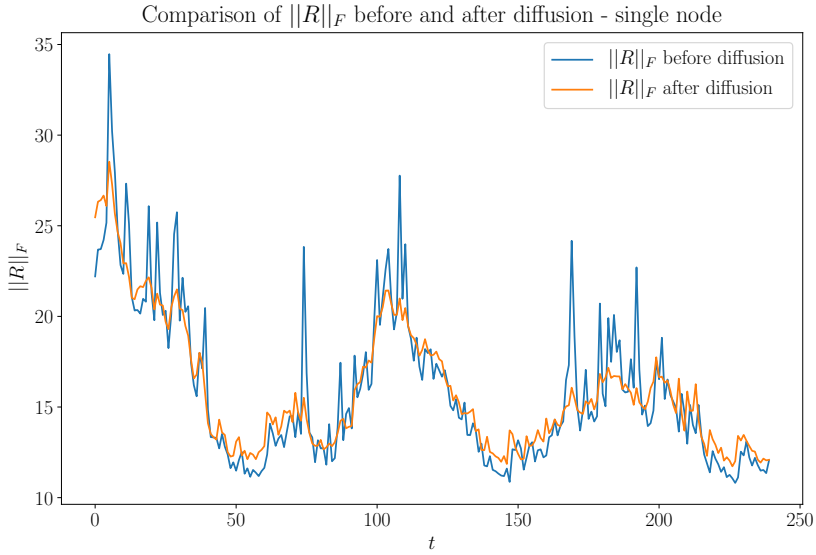


Figure 4.18: The diffusion algorithm smooths the trajectory for this single randomly chosen single node from the network, ignoring significant peaks which would otherwise lead to inferior estimates.

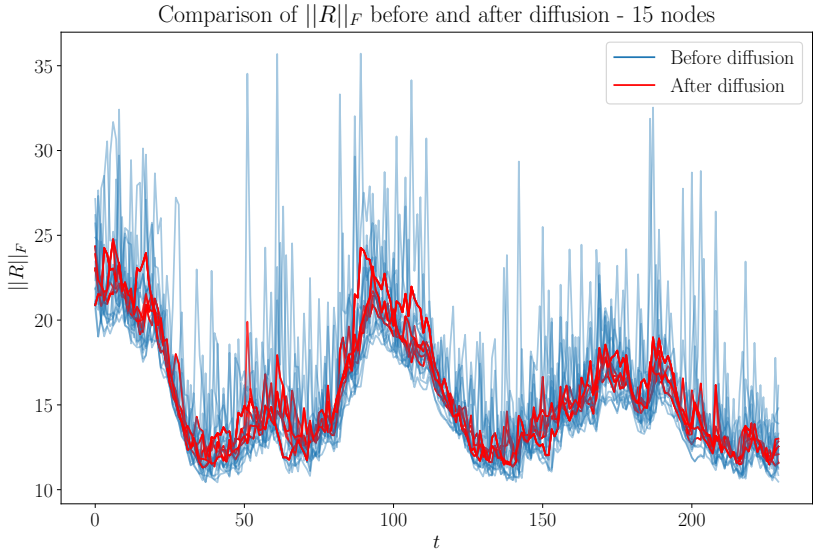


Figure 4.19: All nodes exhibit the behavior of smoothing out the noise covariance estimates, leading to the better results.

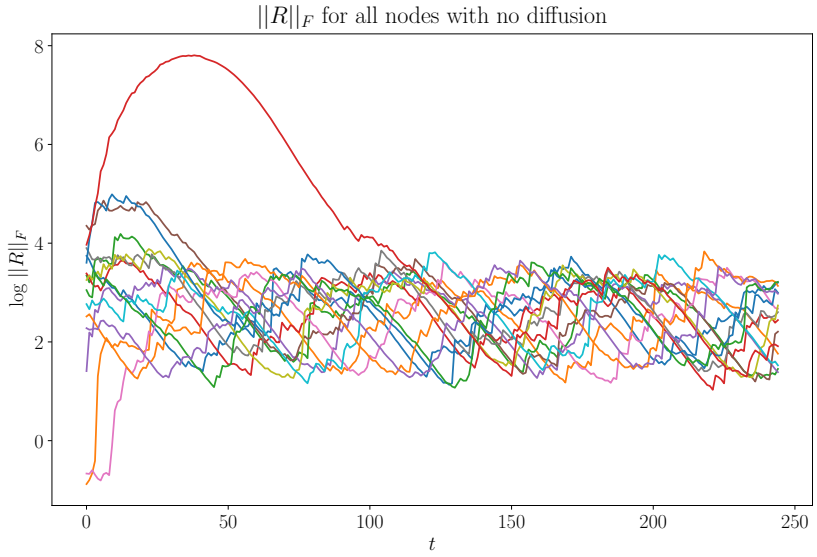


Figure 4.20: Nodes in non-diffusion variant does not exhibit any common pattern. There is also one of the nodes diverging quite significantly, considering the logarithmic scale. The diffusion variant do not exhibit such divergence.

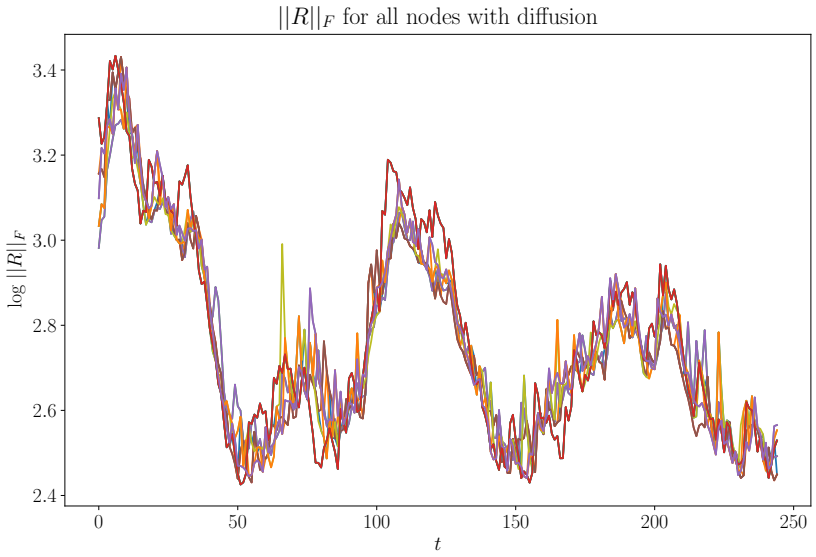


Figure 4.21: Nodes in the diffusion variant are able to pick up the general noise trend and "agree" on it to some degree, even though it may not be the best estimate for the independent nodes.

page 42. While there is almost no or very little pattern in the non-diffused variant, the diffusion variant got a general trend.

It is also interesting that in our simulations, the diffused variance exhibit much fewer divergence cases, which can be explained by the fact that any extremes are much likely to be suppressed by neighbor nodes participating in the diffusion. An example can be seen on Figure 4.22 on page 43.

4.4.4 Conclusion

Indeed, the diffusion extension to the VBAKF algorithm clearly performs better than variant without diffusion in place. This is exhibited consistently over many iterations with different initialization noise. It is relatively robust to the parameter setting when sane defaults are used (see [32] for more information).

The disadvantages of the diffusion algorithm are mostly practical limitations of real-life applications.

First of all, a communication must be established between the nodes. The implementation of this communication may not be possible, nodes can be, for example, only connected to a master node or do not have an ability to receive any information from the outer world, only to send one. Notwithstanding, this is not an intrinsic property of the method, but rather a common requirement for many diffusion and consensus approaches. It is worth to mention that the algorithm does not rely on having all nodes participating in all time steps. As a matter of fact, the worst case scenario is a fallback to the default non-diffused variant when no nodes are communicating. This fact also facilitates the possible implementation in already existing solutions, since it can be implemented gradually, proceeding with the diffusion extension only when it is possible and a preliminary results show its effectiveness.

Secondly, there is additional implementation complexity, but as can be seen from our code, it is very easy to add the diffusion step into the existing KF algorithm. The diffusion strategy

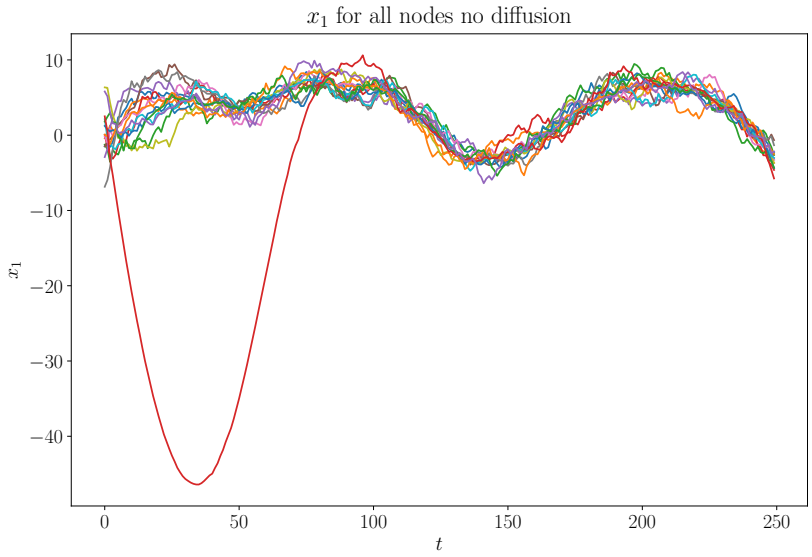


Figure 4.22: An example of diverging node present in the non-diffused variant.

may be of course much more complex than it is in our case, but again, that is not an intrinsic property of the proposed algorithm but rather specific to the use case.

In conclusion, when the node communication is not an issue, we believe that the proposed algorithm can be implemented for a very little price, while not performing worse than a traditional non-diffused variant. Its flexibility allows usage in various scenarios, being able to incorporate arbitrarily complex diffusion strategy.

4.5 Future work

It is obvious that this work does not exhaust the topic in its entirety. To the best of the author's knowledge, there has been a little done in the field of combining parameter diffusion or consensus algorithms with variational Bayes methods. The key questions regarding the algorithm behavior, such as the numerical stability or convergence criteria remains unanswered, unfortunately. Additionally, we believe that it would be a beneficial contribution to the field if at least following were studied:

- What role network's topology plays and how to design networks which are cheap, but still effective?
- What are the most effective diffusion strategies in this setting?
- Sensitiveness to different kinds of (still Gaussian white) noise or its evolution.
- Sensitiveness to different state models.

A higher level extension would be to consider different kinds of white noises when different priors would have to be developed and new equations derived accordingly.

Conclusion

The rise of cheap yet powerful devices equipped with many sensors connected in a communication network justifies the necessity of developing efficient algorithms processing the network's information flow. It is no longer problem of not having the data, but rather generating useful insights based on them.

The algorithm studied in this work is an effective, cheap and elegant improvement of many real-world scenarios where adaptive filtering is being used and noise covariances are unknown and changing. Using exponential family forms, the algorithm is relatively simple to understand and implement even in existing environments, enabling gradual, step-by-step adaptation. It does not suffer from any obvious drawbacks. Its only real disadvantage – a necessity of having a communication protocol over nodes available – is a common requirement for many distributive algorithms and is not an intrinsic property of this one. It is also quite robust to the initial parameters settings and different types of noise, making it an excellent choice in many complex and noisy environments. The nature of the algorithm also enables good introspection and interpretability of the whole model, an important and often overlooked property.

The simulation results strongly support these claims and hopefully offer sufficient explanations for the observed behavior. The author hopes that even though this thesis delivers a rather limited view of the method, it is still a valuable entry point for further research on this matter.

Bibliography

[1] Edwin Thompson Jaynes and G. Larry Bretthorst. *Probability theory: the logic of science*. Cambridge University Press, 2003, p. 727. ISBN: 0521592712.

[2] Alan Hájek. "Interpretations of Probability". In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Winter 201. 2012, pp. 1–4. ISBN: 1095-5054. URL: https://plato.stanford.edu/archives/win2012/entries/probability-interpret/.

[3] Maurice J Dupré and Frank J Tipler. "New Axioms for Rigorous Bayesian Probability". In: *Bayesian Analysis* 1.1 (2004), pp. 1–8. URL: http://dauns.math.tulane.edu/%7B~%7Ddupre/OLDPUBLIC/20090804%20BA%20FNL%203.pdf.

[4] Charles Friedman. "The Frequency Interpretation in Probability". In: *Advances in Applied Mathematics* 23.3 (1999), pp. 234–254. ISSN: 0196-8858. DOI: https://doi.org/10.1006/aama.1999.0653. URL: http://www.sciencedirect.com/science/article/pii/S019688589990653X.

[5] R T Cox. "Probability, Frequency and Reasonable Expectation". In: *American Journal of Physics* 14.1 (1946), pp. 1–13. DOI: 10.1119/1.1990764. URL: https://doi.org/10.1119/1.1990764.

[6] B de Finetti. *Theory of Probability: A critical introductory treatment*. J. Wiley & Sons, Inc., New York, 1975.

[7] E.T. Jaynes. "Bayesian Methods: General Background". In: *Maximum Entropy and Bayesian Methods in Applied Statistics* August 1984 (1986), pp. 1–25. DOI: 10.1017/CBO9780511569678. URL: /chapter.jsf?bid=CBO9780511569678A007%7B%5C&%7Dcid=CBO9780511569678A007.

[8] Alan Stuart and Keith Ord. *Kendall's Advanced Theory of Statistics, Distribution Theory*. Volume 1. Wiley, 2010. ISBN: 978-0470665305.

[9] Dirk P. Kroese and Joshua C.C. Chan. *Statistical modeling and computation*. 2014, pp. 1–400. ISBN: 9781461487753. DOI: 10.1007/978-1-4614-8775-3.

[10] Peter McCullagh. "What is a statistical model?" In: *The Annals of Statistics* 30.5 (2002), pp. 1225–1310. ISSN: 00905364. DOI: 10.1214/aos/1035844977.

[11] Christopher M Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer, 2006. ISBN: 0387310738.

[12] G. Upton and I. Cook. *A Dictionary of Statistics*. 2008, p. 512. ISBN: 978-0-19-954145-4. DOI: 10.1093/acref/9780199541454.001.0001.

[13] Kamil Dedecius and Vladimira Seckarova. "Factorized Estimation of Partially Shared Parameters in Diffusion Networks". In: *IEEE Transactions on Signal Processing* 65.19 (Oct. 2017), pp. 5153–5163. ISSN: 1053-587X. DOI: 10.1109/TSP.2017.2725226. URL: http://ieeexplore.ieee.org/document/7973017/.

[14] David M Blei, Alp Kucukelbir, and Jon D Mcauliffe. "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association* 112.518 (2017). arXiv: arXiv:1601.00670v6. URL: https://arxiv.org/pdf/1601.00670.pdf.

[15] Giorgio Parisi. *Statistical Field Theory*. CRC Press, 1988. ISBN: 978-0738200514.

[16] Greg Welch and Gary Bishop. *An Introduction to the Kalman Filter*. 2006. URL: https://www.cs.unc.edu/%7B~%7Dwelch/media/pdf/kalman%7B%5C_%7Dintro.pdf.

[17] Shih-Tin Lin. "Force Sensing Using Kalman Filtering Techniques for Robot Compliant Motion Control". In: *Journal of Intelligent and Robotic Systems* 18.1 (1997), pp. 1–16. ISSN: 09210296. DOI: 10.1023/A:1007946400645. URL: http://link.springer.com/10.1023/A:1007946400645.

[18] David Gaylor and E. Glenn Lightsey. "GPS/INS Kalman Filter Design for Spacecraft Operating in the Proximity of International Space Station". In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2003. ISBN: 978-1-62410-090-1. DOI: 10.2514/6.2003-5445. URL: http://arc.aiaa.org/doi/10.2514/6.2003-5445.

[19] Gurnain Kaur Pasricha. "Kalman Filter and its Economic Applications". In: *Munich Personal RePEc Archive* (2006). URL: http://mpra.ub.uni-muenchen.de/22734/.

[20] Ingvar Strid and Karl Walentin. "Block Kalman Filtering for Large-Scale DSGE Models". In: *Computational Economics* 33.3 (Apr. 2009), pp. 277–304. ISSN: 0927-7099. DOI: 10.1007/s10614-008-9160-4. URL: http://link.springer.com/10.1007/s10614-008-9160-4.

[21] S. Gannot. "Speech processing utilizing the Kalman filter". In: *IEEE Instrumentation & Measurement Magazine* 15.3 (June 2012), pp. 10–14. ISSN: 1094-6969. DOI: 10.1109/MIM.2012.6204866. URL: http://ieeexplore.ieee.org/document/6204866/.

[22] Daniel M. Wolpert and Zoubin Ghahramani. "Computational principles of movement neuroscience". In: *Nature Neuroscience* 3 (Nov. 2000), pp. 1212–1217. ISSN: 10976256. DOI: 10.1038/81497. URL: http://www.nature.com/doifinder/10.1038/81497.

[23] Magdi S. Mahmoud and Haris M. Khalid. "Distributed Kalman filtering: a bibliographic review". In: *IET Control Theory & Applications* 7.4 (2013), pp. 483–501. DOI: 10.1049/iet-cta.2012.0732. URL: http://cogprints.org/8906/1/MsM-KFUPM-DCC-2D-One%7B%5C%7D5B3R%7B%5C%7D5D.pdf.

[24] Felix Govaers. "Distributed Kalman Filter". In: *Kalman Filters - Theory for Advanced Applications*. InTech, Feb. 2018. DOI: 10.5772/intechopen.71941. URL: http://www.intechopen.com/books/kalman-filters-theory-for-advanced-applications/distributed-kalman-filter.

[25] Ian F. Akyildiz and Mehmet Can Vuran. *Wireless Sensor Networks*. 2010, p. 516. ISBN: 9780470515181. DOI: 10.1002/9780470515181. URL: http://scholar.google.com/scholar?hl=en%7B%5C&%7DbtnG=Search%7B%5C&%7Dq=intitle:No+Title%7B%5C#%7D0%7B%5C%7D5Cnhttp://doi.wiley.com/10.1002/9780470515181.

[26] G. Portz et al. "Field comparison of ultrasonic and canopy reflectance sensors used to estimate biomass and N-uptake in sugarcane". In: *Stafford J.V. (eds)* (2013), pp. 111–117. DOI: 10.3920/978-90-8686-778-3_12. URL: https://link.springer.com/chapter/10.3920%7B%5C%7D2F978-90-8686-778-3%7B%5C_%7D12.

[27] Mark E. Campbell and Nisar R. Ahmed. "Distributed Data Fusion: Neighbors, Rumors, and the Art of Collective Knowledge". In: *IEEE Control Systems* 36.4 (Aug. 2016), pp. 83–109. ISSN: 1066-033X. DOI: 10.1109/MCS.2016.2558444. URL: http://ieeexplore.ieee.org/document/7515322/.

[28] Dan Simon. *No Title*. Hoboken, NJ, USA: John Wiley & Sons, Inc., May 2006, p. 526. ISBN: 9780470045343. DOI: 10.1002/0470045345. URL: http://doi.wiley.com/10.1002/0470045345.

[29] R. Mehra. "Approaches to adaptive filtering". In: *IEEE Transactions on Automatic Control* 17.5 (Oct. 1972), pp. 693–698. ISSN: 0018-9286. DOI: 10.1109/TAC.1972.1100100. URL: http://ieeexplore.ieee.org/document/1100100/.

[30] C. Hide, T. Moore, and M. Smith. "Adaptive Kalman filtering algorithms for integrating GPS and low cost INS". In: *PLANS 2004. Position Location and Navigation Symposium (IEEE Cat. No.04CH37556)*. IEEE, pp. 227–233. ISBN: 0-7803-8416-4. DOI: 10.1109/PLANS.2004.1308998. URL: http://ieeexplore.ieee.org/document/1308998/.

[31] Myeong-Jong Yu. "INS/GPS Integration System using Adaptive Filter for Estimating Measurement Noise Variance". In: *IEEE Transactions on Aerospace and Electronic Systems* 48.2 (2012), pp. 1786–1792. ISSN: 0018-9251. DOI: 10.1109/TAES.2012.6178100. URL: http://ieeexplore.ieee.org/document/6178100/.

[32] Yulong Huang et al. "A Novel Adaptive Kalman Filter With Inaccurate Process and Measurement Noise Covariance Matrices". In: *IEEE Transactions on Automatic Control* 63.2 (Feb. 2018), pp. 594–601. ISSN: 0018-9286. DOI: 10.1109/TAC.2017.2730480. URL: http://ieeexplore.ieee.org/document/8025799/.

[33] A. P. Sage and G. W. Husa. "Adaptive Filtering with Unknown Prior Statistics". In: *National Technical Reports Library* (1969).

[34] Xiangdong Gao, Deyong You, and Seiji Katayama. "Seam Tracking Monitoring Based on Adaptive Kalman Filter Embedded Elman Neural Network During High-Power Fiber Laser Welding". In: *IEEE Transactions on Industrial Electronics* 59.11 (Nov. 2012), pp. 4315–4325. ISSN: 0278-0046. DOI: 10.1109/TIE.2012.2193854. URL: http://ieeexplore.ieee.org/document/6179988/.

[35] Maja Karasalo and Xiaoming Hu. "An optimization approach to adaptive Kalman filtering". In: *Automatica* 47.8 (Aug. 2011), pp. 1785–1793. ISSN: 0005-1098. DOI: 10.1016/J.AUTOMATICA.2011.04.004. URL: https://www.sciencedirect.com/science/article/pii/S000510981100224X.

[36] X.R. Li and Y. Bar-Shalom. "A recursive multiple model approach to noise identification". In: *IEEE Transactions on Aerospace and Electronic Systems* 30.3 (July 1994), pp. 671–684. ISSN: 00189251. DOI: 10.1109/7.303738. URL: http://ieeexplore.ieee.org/document/303738/.

[37] Federico S. Cattivelli and Ali H. Sayed. "Diffusion Strategies for Distributed Kalman Filtering and Smoothing". In: *IEEE Transactions on Automatic Control* 55.9 (Sept. 2010), pp. 2069–2084. ISSN: 0018-9286. DOI: 10.1109/TAC.2010.2042987. URL: http://ieeexplore.ieee.org/document/5411741/.

[38] Kamil Dedecius and Petar M. Djuric. "Sequential Estimation and Diffusion of Information Over Networks: A Bayesian Approach With Exponential Family of Distributions". In: *IEEE Transactions on Signal Processing* 65.7 (Apr. 2017), pp. 1795–1809. ISSN: 1053-587X. DOI: 10.1109/TSP.2016.2641380. URL: http://ieeexplore.ieee.org/document/7790804/.

[39] G. Maurice Kendall et al. *Kendall's Advanced Theory of Statistics, volume 2B: Bayesian Inference, second edition*. Edward Arnold, 2004. ISBN: 0340807520. URL: https://eprints.soton.ac.uk/46376/.

[40] Simo Särkkä Jouni Hartikainen. *Variational Bayesian Adaptation of Noise Covariances in Non-Linear Kalman Filtering*. Feb. 2013. arXiv: 1302.0681. URL: http://arxiv.org/abs/1302.0681.