Dissertation for the Faculty of Computer Science (FIN),
Otto von Guericke University (OVGU), Magdeburg

Data Confidentiality for Distributed Sensor Fusion

Marko Ristic

January 17, 2023

Reviewers:

Prof. Dr.-Ing. Benjamin Noack (OVGU, Magdeburg, Germany)

. . .

Contents

Acknowledgements iii

Abstract iv

Kurzfassung v

Notation vi

1. Introduction 1

1.1. State-of-the-Art and Research Questions 2

1.1.1. Confidential Estimate Fusion 3

1.1.2. Confidential Distributed Non-Linear Measurement Models 3

1.1.3. Provable Estimate Performances 3

1.2. Contributions 3

1.3. Thesis Structure 3

2. Preliminaries 4

2.1. Estimation Preliminaries 4

2.1.1. Kalman Filter 4

2.1.2. Kalman Filter Optimality 5

2.1.3. Extended Kalman Filter 5

2.1.4. Information Filter 6

2.1.5. Extended Information Filter 8

2.1.6. Covariance Intersection and Fast Covariance Intersection 9

2.2. Encryption Preliminaries 10

2.2.1. Meeting Cryptographic Notions 11

2.2.2. Paillier Homomorphic Encryption Scheme 12

2.2.3. Joye-Libert Aggregation Scheme 12

2.2.4. Lewi Order-Revealing Encryption Scheme 13

2.2.5. Encoding Numbers for Encryption 14

3. Estimate Fusion on an Untrusted Cloud 16

3.1. Problem Formulation 16

3.2. Confidential Cloud Fusion Leaking Fusion Weights 16

3.2.1. Secure Fast Covariance Intersection 16

3.2.2. Computational Complexity 21

3.2.3. Security Analysis 22

3.2.4. Simulation 22

3.3. Confidential Cloud Fusion Without Leaking Fusion Weights 23
3.3.1. Computational Complexity 26
3.3.2. Security Analysis 26
3.3.3. Simulation 27
3.4. Conclusions on Confidential Estimate Fusion 28

4. Distributed Non-Linear Measurement Fusion with Untrusted Participants 29
4.1. Problem Formulation 29
4.1.1. Formal Cryptographic Problem 29
4.1.2. Estimation problem 32
4.2. Confidential Range-Only Localisation 33
4.2.1. Private Linear Combination Aggregation Scheme 33
4.2.2. Privacy-Preserving Localisation 35
4.2.3. Solvable Sub-Class of Non-Linear Measurement Models 43
4.2.4. Security Analysis 43
4.2.5. Simulation 43
4.3. Conclusions on Confidential Distributed Non-Linear Measurement Fusion 45

5. Provable Estimation Performances 48
5.1. Problem Formulation 48
5.1.1. Formal Cryptographic Problem 48
5.1.2. Estimation Problem 50
5.1.3. Multi-sensor Problem 50
5.2. Privileged Estimation for Linear Systems 51
5.2.1. Gaussian Keystream 51
5.2.2. Measurement Modification 52
5.2.3. Multiple Privileges 53
5.2.4. Security Analysis 54
5.2.5. Simulation 57
5.3. Fusion in Privileged Estimation Environments 59
5.3.1. Noise Generation 59
5.3.2. Measurement Modification 60
5.3.3. Distribution of Noise Terms 63
5.3.4. Security Analysis 63
5.3.5. Simulation 67
5.4. Conclusions on Provable Estimation Performances 70

6. Conclusion 71

A. Linear-Combination Aggregator Obliviousness 72

B. Cryptographic Proof of LCAO Scheme Security 73

Acknowledgements

Acks go here.

Abstract

Distributed sensing and fusion algorithms are increasingly present in public computing networks and have led to a natural concern for data security in these environments. This thesis aims to present generalisable data fusion algorithms that simultaneously provide strict cryptographic guarantees on user data confidentiality. While fusion algorithms providing some degrees of security guarantees exist, these are typically either provided at the cost of solution generality or lack formal security proofs. Here, novel cryptographic constructs and state-of-the-art encryption schemes are used to develop formal security guarantees for new and generalised data fusion algorithms. Industry-standard Kalman filter derivates are modified and existing schemes abstracted such that novel cryptographic notions capturing the required communications can be formalised, while simulations provide an analysis of practicality. Due to the generality of the presented solutions, broad applications are supported, including autonomous vehicle communications, smart sensor networks and distributed localisation.

Kurzfassung

German abs go here.

Notation

Complete the notation here.

1. Introduction

Sensor data processing, state estimation and data fusion have long been active areas of research and continue to find applications in modern systems [1, 2]. As distributed networks have become more prevalent over the years, greater stress has been put on the need for broadly applicable algorithms that support varying types of measurements, estimate accuracies and communication availabilities [3, 4], finding uses in localisation, weather forecasting, mapping, cooperative computing and cloud computing []. The use of Bayesian estimation methods such as the popular Kalman filter and its non-linear derivatives have become especially prevalent in application due to their recursive, often optimal, estimation properties and their suitability for modelling cross-correlations between local estimates [5, 6]. The handling of these cross-correlations, especially when they are not known in advance, is a common difficulty in state estimation and is tied to the challenges within the field [7]. The methods presented in this thesis and much of the related work in data fusion tasks similarly require the consideration of these challenges.

While the challenges relating to correlation errors are a well-established field of research, widespread advancements in distributed computing and uses of public networks for sensor communication have put a focus on the additional requirements of data privacy and state secrecy in recent years as well [8, 9]. Problems that require both the handling of correlation errors and guaranteeing a level of security for the participants involved are therefore a relevant topic in state estimation today and at the core of the work presented in this thesis. Achieving cryptographic secrecy typically involves hiding transferred information from unauthorised parties and can often be achieved irrespective of the estimation algorithms used by using common symmetric and public-key encryption schemes such as the Advanced Encryption Standard (AES) [10] and the Rivest-Shamir-Adleman cryptosystem (RSA) [11], respectively. These scenarios, however, imply a trust between encrypting and decrypting parties, which cannot always be assumed in distributed environments. In addition, partial computations on encrypted data or the intended leakage of some results are sometimes required for computing final results. This has led to several operation-providing and leakage-supporting encryption schemes [12, 13, 14, 15, 16] suitable for these distributed environments or when fine control over leakage is required. While these cryptographic schemes and notations are applicable in estimation and fusion tasks, the nature of cryptographic analysis in distributed environments, heavily dependent on communication protocols, has meant that developed solutions are often very context-specific, leading to numerous solutions for various estimation scenarios, use-cases and security requirements. This leads us to the current state-of-the-art literature on security-oriented state estimation and data fusion, observable gaps in this literature and the research questions we aim to answer.

1.1. State-of-the-Art and Research Questions

To summarise relevant work in security-oriented state estimation and data fusion we first discuss the security and types of data explored in this thesis. Regarding algorithms, we primarily consider stochastic models and outputs. As discussed in section [], Bayesian estimation methods such as the Kalman filter are prevalent due to their applicability and suitability for modelling phenomena accurately. In terms of security, we restrict ourselves to the data confidentiality component of the Confidentiality-Integrity-Availability (CIA) triad []. That is, the primary concern of security in this thesis is that concrete data deemed to be private to some participants remain so and its leakage is formally quantifiable. Data privacy, a related concept, is concerned with stopping the identification of individuals from available information. Although data privacy encompasses data confidentiality, it is a broader topic including communication traffic analyses and external cross-referencing to identify individuals and is not considered in its entirety in this thesis. That said, the terms *data privacy* and *privacy-preserving* are often used in the state-of-the-art to refer to data confidentiality alone and the ability to identify individuals from concrete data available []. The terms will be similarly used throughout this thesis.

Since knowing exact communications between participating parties is required for meaningful cryptographic analysis of transferred data, many existing general estimation algorithms have been restricted in some way to make communication and security easier to discuss. For example, [aristov] presents a distributed Kalman filter, namely an Information filter, where sensor measurements and measurement errors are known only to the measuring sensors while final estimates are leaked to an estimator. The restriction for this to be achieved requires sensors to form a hierarchical communication structure and measurement models to be linear, limiting the otherwise broadly applicable non-linear models or arbitrary communication structures. Another work, [proloc], presents localisation using range-only measurements where measurements and sensor locations, both required for localisation, are kept private to sensors and a centralised estimator while final estimates are available to an external trusted party. Here, no communication structure is enforced but produced estimates provide no error statistics and do not consider a dynamic process model.

In [pwsac, pwsah],

aggregation papers

differential privacy and differentially private Kalman filtering

privacy-preserving optimisation with security based on statistical estimation

added noise estimation

–

privacy-preserving image-based localisation

eavesdropper paper with a secure return channel and a lossier channel for eavesdroppers

GPS

chaotic system paper

physical layer noise paper (similar to chaotic noise paper)

–

The two different approaches, restricting existing broad estimation methods in some ways to make cryptographic analysis plausible and ignoring formal security when assumptions and conclusions are intuitive demonstrate a gap in the existing literature and bring us to the target research topics this thesis aims to explore.

These broad topics aim to fulfil the goal of generalisable but cryptographically provable estimation and fusion methods in distributed environments and lead to the concrete problems tackled in this work

1.1.1. Confidential Estimate Fusion

1.1.2. Confidential Distributed Non-Linear Measurement Models

1.1.3. Provable Estimate Performances

1.2. Contributions

The contributions tackle the research topics in section .. by considering three concrete problems that coincide with the broader problems in the field

- dot point topics

1.3. Thesis Structure

Each chapter includes relevant related literature to the specific problem and a formal problem formalisation before presenting the novel solutions.

2. Preliminaries

When introducing novel methods throughout this thesis, we make use of several existing algorithms and constructs. In this chapter, we present these relevant preliminaries grouped by the fields they pertain to; estimation and cryptography.

2.1. Estimation Preliminaries

Sensor and estimate data that we consider is primarily Bayesian in nature and typically consists of estimates and associated estimate uncertainties. The linear Kalman filter and the linearising extended Kalman filter, along with their information filter equivalents, are particularly useful in the estimation and fusion of such data. A general fusion algorithm, the covariance intersection, used when data cross-correlations are unknown, is also introduced.

2.1.1. Kalman Filter

The Kalman filter (KF) [orig kf] is a popular and well studied recursive state estimation filter that produces estimates and their error covariances $\hat{\underline{x}}_{k|k'} \in \mathbb{R}^n$ and $\mathbf{P}_{k|k'} \in \mathbb{R}^{n\times n}$, respectively, for a timestep $k \in \mathbb{N}$, given measurements up to and including timestep $k' \in \mathbb{N}$ [kf uses]. Although the KF supports the estimation of a system state which can be manipulated through an external input, this thesis primarily discusses scenarios where no external inputs are known to the estimator and will introduce the filter with these inputs set to $\underline{0}$. In this form, the KF assumes the existence of a true state $\underline{x}_k \in \mathbb{R}^n$ at each timestep k, following the linear process model

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

where $\underline{w}_k \sim \mathcal{N}(\underline{0}, \mathbf{Q}_k)$ with known covariance $\mathbf{Q}_k \in \mathbb{R}^{n\times n}$. Similarly, measurements $\underline{z}_k \in \mathbb{R}^m$ are assumed to follow the linear measurement model

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

where $\underline{v}_k \sim \mathcal{N}(\underline{0}, \mathbf{R}_k)$ with known covariance $\mathbf{R}_k \in \mathbb{R}^{m\times m}$. The filter requires initialisation with some known values $\hat{\underline{x}}_{0|0}$ and $\mathbf{P}_{0|0}$ and is computed recursively in two steps. First, the estimate for the next timestep is predicted without new measurement information, known as the *prediction* step, and is given by

$$\hat{\underline{x}}_{k|k-1} = \mathbf{F}_k \hat{\underline{x}}_{k-1|k-1} \tag{2.3}$$

and

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^\top + \mathbf{Q}_k \,. \tag{2.4}$$

Next, this prediction is updated with current measurement information, known as the *update* step, and given by

$$\underline{\hat{x}}_{k|k} = \underline{\hat{x}}_{k|k-1} + \mathbf{P}_{k|k-1} \mathbf{H}_k^\top \left(\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R}_k \right)^{-1} \left(z_k - \mathbf{H}_k \underline{\hat{x}}_{k|k-1} \right) \tag{2.5}$$

and

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1} \mathbf{H}_k^\top \left(\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R}_k \right)^{-1} \mathbf{H} \mathbf{P}_{k|k-1} \,. \tag{2.6}$$

In addition to alternating prediction and update steps as time progresses, the update step (2.5) and (2.6) can be skipped at timesteps when no measurements are available. Similarly, when multiple independent measurements are present at the same timestep, the update step can be repeated for each measurement individually. Detailed derivations of the KF and discussions on its properties can be found in [huag+chap].

2.1.2. Kalman Filter Optimality

One of the reasons for the ubiquity and popularity of the KF introduced in section 2.1.1 is its optimality in terms of mean square error (MSE) [huag+chap]. That is, the estimate's error covariances, defined by the expectation capturing mean square error,

$$\mathbf{P}_{k|k} = \mathbb{E}\left\{ \left(\underline{x}_k - \underline{\hat{x}}_{k|k} \right) \left(\underline{x}_k - \underline{\hat{x}}_{k|k} \right)^\top \right\} \,, \tag{2.7}$$

and computed by (2.4) and (2.6), can be shown to equal the theoretical lower bound on the covariance of an unbiased estimator when process and measurement models (2.1) and (2.2), respectively, capture the estimated environment exactly [huag refs for crlb]. This property will be used in later cryptographic discussions in this thesis to guarantee estimator performances in terms of MSE. Further reading on the definitions and proofs of KF optimality can be found in [crlb,huag,etc].

2.1.3. Extended Kalman Filter

The extended Kalman filter (EKF) is a recursive state estimation filter applicable to non-linear models and closely related to the linear KF [ekf paper,huag+chap]. The filter produces estimates and their covariances at each timestep by linearising models at the current estimate and evaluating the filter similarly to the KF. As in the KF, a true state $\underline{x}_k$ is assumed to follow known models. The process model is now non-linear and given by

$$\underline{x}_k = f_k(\underline{x}_{k-1}) + \underline{w}_k \,, \tag{2.8}$$

where again $\underline{w}_k \sim \mathcal{N}(\underline{0}, \mathbf{Q}_k)$ with known covariance $\mathbf{Q}_k$. Similarly, measurements are assumed to follow the non-linear measurement model

$$\underline{z}_k = h_k(\underline{x}_k) + \underline{v}_k \,, \tag{2.9}$$

with $\underline{v}_k \sim \mathcal{N}(\underline{0}, \mathbf{R}_k)$ and known covariance $\mathbf{R}_k$. The EKF *prediction* step is given by

$$\hat{\underline{x}}_{k|k-1} = f_k\left(\hat{\underline{x}}_{k-1|k-1}\right) \tag{2.10}$$

and

$$\mathbf{P}_{k|k-1} = \hat{\mathbf{F}}_k \mathbf{P}_{k-1|k-1} \hat{\mathbf{F}}_k^\top + \mathbf{Q}_k \,, \tag{2.11}$$

with Jacobian

$$\hat{\mathbf{F}}_k = \left.\frac{\partial f_k}{\partial \underline{x}}\right|_{\hat{\underline{x}}_{k-1|k-1}} \tag{2.12}$$

linearising the process model at the latest estimate for estimate error covariance prediction. The EKF *update* step is given by

$$\hat{\underline{x}}_{k|k} = \hat{\underline{x}}_{k|k-1} + \mathbf{P}_{k|k-1}\hat{\mathbf{H}}_k^\top \left(\hat{\mathbf{H}}_k \mathbf{P}_{k|k-1} \hat{\mathbf{H}}_k^\top + \mathbf{R}_k\right)^{-1} \left(z_k - h_k(\hat{\underline{x}}_{k|k-1})\right) \tag{2.13}$$

and

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1}\hat{\mathbf{H}}_k^\top \left(\hat{\mathbf{H}}_k \mathbf{P}_{k|k-1} \hat{\mathbf{H}}_k^\top + \mathbf{R}_k\right)^{-1} \hat{\mathbf{H}} \mathbf{P}_{k|k-1} \,, \tag{2.14}$$

with Jacobian

$$\hat{\mathbf{H}}_k = \left.\frac{\partial h_k}{\partial \underline{x}}\right|_{\hat{\underline{x}}_{k|k-1}} \tag{2.15}$$

linearising the measurement model. Unlike the linear KF, by linearising the models the EKF propagates Gaussian model noises in its estimates that may not be Gaussian in reality, even when process and measurement models (2.8) and (2.9), respectively, are exactly correct. For this reason, the EKF does not hold the same guarantees on optimality as the KF does. To a similar effect, highly non-linear models or inaccurate models can lead to greater inaccuracies and divergence of estimates from true states in EKF estimates. Despite these downsides, its scalability and efficiency have made the EKF an industry-standard estimation filter for non-linear systems [smth on uses of ekf]. More details on the EKF and its derivation can be found in [huag+chap].

2.1.4. Information Filter

The information filter (IF) is an algebraic reformulation of the KF from section 2.1.1 [niehsenInformationFusionBased2002, Mutambara, another]. The information form of the filter simplifies the update step of the filter making it more suitable when multiple independent measurements are present at the same timestep. The key difference of the IF is the storing and propagation of the information vector $\hat{\underline{y}}_{k|k'}$ and information matrix $\mathbf{Y}_{k|k'}$ rather than the estimate and its error covariance, $\hat{\underline{x}}_{k|k'}$ and $\mathbf{P}_{k|k'}$, respectively, stored by the KF. When assuming the same linear and Gaussian models (2.1) and (2.2), the information vector and matrix are related to the estimate and its covariance by

$$\underline{\hat{y}}_{k|k'} = \mathbf{P}_{k|k'}^{-1}\underline{\hat{x}}_{k|k'} \tag{2.16}$$

and

$$\mathbf{Y}_{k|k'} = \mathbf{P}_{k|k'}^{-1}\,, \tag{2.17}$$

for an estimated timestep k and measurements from timesteps up to and including k'. The estimation of the information vector and information matrix requires an initialisation of $\underline{\hat{y}}_{0|0}$ and $\mathbf{Y}_{0|0}$, similarly to the KF, and is also performed by iterating distinct predict and update filter steps. The prediction step is given by

$$\underline{\hat{y}}_{k|k-1} = \mathbf{Y}_{k|k-1}\mathbf{F}_k\mathbf{Y}_{k-1|k-1}^{-1}\underline{\hat{y}}_{k-1|k-1} \tag{2.18}$$

and

$$\mathbf{Y}_{k|k-1} = \left(\mathbf{F}_k\mathbf{Y}_{k-1|k-1}^{-1}\mathbf{F}_k^{\top} + \mathbf{Q}_k\right)^{-1}\,. \tag{2.19}$$

The update step is given by

$$\underline{\hat{y}}_{k|k} = \underline{\hat{y}}_{k|k-1} + \underline{i}_k \tag{2.20}$$

and

$$\mathbf{Y}_{k|k} = \mathbf{Y}_{k|k-1} + \mathbf{I}_k\,, \tag{2.21}$$

where added terms $\underline{i}_k$ and $\mathbf{I}_k$ are known as the measurement vector and measurement matrix, respectively, and are defined as

$$\underline{i}_k = \mathbf{H}_k^{\top}\mathbf{R}_k^{-1}z_k \tag{2.22}$$

and

$$\mathbf{I}_k = \mathbf{H}_k^{\top}\mathbf{R}_k^{-1}\mathbf{H}_k\,. \tag{2.23}$$

Since all information related to a measurement and its sensor are captured in $\underline{i}_k$ and $\mathbf{I}_k$, namely the measured value z_k, measurement model $\mathbf{H}_k$ and measurement error $\mathbf{R}_k$, sequential IF update steps required in the presence of multiple sensors are easily computed as a summation of this information from each sensor. That is, if we consider the same process model (2.1) and multiple sensors i, $1 \leq i \leq n$, making independent measurements that follow models

$$\underline{z}_{k,i} = \mathbf{H}_{k,i}\underline{x}_k + \underline{v}_{k,i}\,, \tag{2.24}$$

with $\underline{v}_{k,i} \sim \mathcal{N}(\underline{0}, \mathbf{R}_{k,i})$ and known covariances $\mathbf{R}_{k,i}$, the update step of the filter using all measurements at timestep k can be written as

$$\underline{\hat{y}}_{k|k} = \underline{\hat{y}}_{k|k-1} + \sum_{i=1}^{n}\underline{i}_{k,i} \tag{2.25}$$

and

$$\mathbf{Y}_{k|k} = \mathbf{Y}_{k|k-1} + \sum_{i=1}^{n} \mathbf{I}_{k,i} \,, \tag{2.26}$$

where information vectors $\underline{i}_{k,i}$ and information matrices $\mathbf{I}_{k,i}$ are now dependent on sensor i and given by

$$\underline{i}_{k,i} = \mathbf{H}_{k,i}^{\top}\mathbf{R}_{k,i}^{-1}\underline{z}_{k,i} \tag{2.27}$$

and

$$\mathbf{I}_{k,i} = \mathbf{H}_{k,i}^{\top}\mathbf{R}_{k,i}^{-1}\mathbf{H}_{k,i} \,. \tag{2.28}$$

The easily computed summation has led to the IF being particularly suited to distributed estimation environments, where multiple sensors are present and communicational costs need to be reduced [if egs]. In addition, since the IF is strictly a rearrangement of terms in the KF, it holds the same optimality properties as the KF, described in section 2.1.2. For additional reading on the IF, see [mutambara+chap].

2.1.5. Extended Information Filter

The extended information filter (EIF) is an algebraic reformulation of the EKF and represents a non-linear model extension to the linear IF [thrun, another]. As with the IF, its simplification of the update step to a trivial sum has led to its adoption in suitable distributed environments with multiple sensors and a desire to reduce communicational costs [tobias garritsen thesis]. Similarly, the EIF estimates and propagates the information vector $\hat{\underline{y}}_{k|k'}$ and information matrix $\mathbf{Y}_{k|k'}$ that relate to the state estimate and its covariance by (2.16) and (2.17), respectively. Assuming the non-linear process model (2.8) and multiple sensors i, $1 \leq i \leq n$, making independent non-linear measurements that follow models

$$\underline{z}_{k,i} = h_{k,i}\left(\underline{x}_k\right) + \underline{v}_{k,i} \,, \tag{2.29}$$

with $\underline{v}_{k,i} \sim \mathcal{N}(\underline{0}, \mathbf{R}_{k,i})$ and known covariances $\mathbf{R}_{k,i}$, the EIF predict step is given by

$$\hat{\underline{y}}_{k|k-1} = \mathbf{Y}_{k|k-1} f_k\left(\mathbf{Y}_{k-1|k-1}^{-1}\hat{\underline{y}}_{k-1|k-1}\right) \tag{2.30}$$

and

$$\mathbf{Y}_{k|k-1} = \left(\hat{\mathbf{F}}_k \mathbf{Y}_{k-1|k-1}^{-1}\hat{\mathbf{F}}_k^{\top} + \mathbf{Q}_k\right)^{-1} \,, \tag{2.31}$$

with Jacobian linearising the process model

$$\hat{\mathbf{F}}_k = \left.\frac{\partial f_k}{\partial \underline{x}}\right|_{\hat{\underline{x}}_{k-1|k-1}} \,. \tag{2.32}$$

The update step of the filter using all n measurements is given by

$$\hat{\underline{y}}_{k|k} = \hat{\underline{y}}_{k|k-1} + \sum_{i=1}^{n} \underline{i}_{k,i} \tag{2.33}$$

and

$$\mathbf{Y}_{k|k} = \mathbf{Y}_{k|k-1} + \sum_{i=1}^{n} \mathbf{I}_{k,i} \,, \tag{2.34}$$

where information vectors $\underline{i}_{k,i}$ and information matrices $\mathbf{I}_{k,i}$ now linearise the measurement model and are given by

$$\underline{i}_{k,i} = \hat{\mathbf{H}}_{k,i}^{\top} \mathbf{R}_{k,i}^{-1} \left(z_{k,i} - h_{k,i} \left(\mathbf{Y}_{k|k-1}^{-1} \hat{\underline{y}}_{k|k-1} \right) + \hat{\mathbf{H}}_{k,i} \mathbf{Y}_{k|k-1}^{-1} \hat{\underline{y}}_{k|k-1} \right) \tag{2.35}$$

and

$$\mathbf{I}_{k,i} = \hat{\mathbf{H}}_{k,i}^{\top} \mathbf{R}_{k,i}^{-1} \hat{\mathbf{H}}_{k,i} \,, \tag{2.36}$$

with Jacobian

$$\hat{\mathbf{H}}_{k,i} = \left. \frac{\partial h_{k,i}}{\partial \underline{x}} \right|_{\hat{\underline{x}}_{k|k-1}} . \tag{2.37}$$

Similarly to the EKF, the linearisation of models leads to estimation errors making optimality guarantees of the KF and IF not hold for the EIF. For further reading and applications of the EIF, see [refs].

2.1.6. Covariance Intersection and Fast Covariance Intersection

In the filters presented in the previous sections, measurements from the same timestep must be independent for sequential update steps to fuse them correctly. That is, non-zero cross-correlations between measurements can lead to overly confident estimate error covariances and estimation track divergence [crosscor paper]. In some cases, this independence of measurements cannot be guaranteed and a conservative fusion method is required to obtain a final estimate and its error covariance using all available measurements. A typical scenario is the fusion of local estimator estimates themselves, such as those produced by separate KF or EKF instances, that may contain cross-correlations due to shared process model assumptions during estimation [ci and dependence ref]. Covariance intersection (CI), is a fusion algorithm that fuses estimates and their error covariances (or measurements and their known error covariances) when cross-correlations are unknown [julierNondivergentEstimation] and produces estimates that are guaranteed to be conservative, that is, not overly confident in the error of resulting fused estimates. The CI fusion of n estimates $\hat{\underline{x}}_i$, $1 \leq i \leq n$, and their error covariances $\mathbf{P}_i$, $1 \leq i \leq n$, produces a fused estimate $\hat{\underline{x}}_{\textsf{fus}}$ and its error covariance $\mathbf{P}_{\textsf{fus}}$, and is given by

$$\hat{\underline{x}}_{\textsf{fus}} = \mathbf{P}_{\textsf{fus}} \sum_{i=1}^{n} \omega_i \mathbf{P}_i^{-1} \hat{\underline{x}}_i \tag{2.38}$$

and

$$\mathbf{P}_{\textsf{fus}} = \left(\sum_{i=1}^{n} \omega_i \mathbf{P}_i^{-1} \right)^{-1} , \tag{2.39}$$

for some weights $0 \leq \omega_i \leq 1$, $1 \leq i \leq n$, such that

$$\sum_{i=1}^{n} \omega_i = 1 \,. \tag{2.40}$$

The weights ω_i are chosen in a way to speed up the convergence of estimate errors over time and to minimise fusion estimate error by minimising some property of the fused estimate error covariance. A common choice is to minimise the trace of the covariance [ci example with trace], requiring a solution to

$$\underset{\omega_1,\ldots,\omega_n}{\arg\min} \left\{ \operatorname{tr}\left(\mathbf{P}_{\mathsf{fus}}\right) \right\} \,. \tag{2.41}$$

However, minimising the non-linear cost function (2.41) can be computationally costly and has led to the development of faster approximation techniques. The Fast Covariance Intersection (FCI) algorithm [niehsenInformationFusionBased] is one such method, that approximates (2.41) non-iteratively, while still guaranteeing consistency. It is defined by adding new constraints

$$\omega_i \operatorname{tr}(\mathbf{P}_i) - \omega_{i+1} \operatorname{tr}(\mathbf{P}_{i+1}) = 0 \,, \tag{2.42}$$

for $1 \leq i \leq n-1$, leading to the linear problem

$$\begin{bmatrix} \mathcal{P}_1 & -\mathcal{P}_2 & 0 & \cdots & 0 \\ 0 & \mathcal{P}_2 & -\mathcal{P}_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathcal{P}_{n-1} & -\mathcal{P}_n \\ 1 & \cdots & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_{n-1} \\ \omega_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{2.43}$$

and its solution

$$\omega_i = \frac{\frac{1}{\mathcal{P}_i}}{\sum_{j=1}^{n} \frac{1}{\mathcal{P}_j}} \,, \tag{2.44}$$

for $1 \leq i \leq n$, where $\mathcal{P}_i = \operatorname{tr}(\mathbf{P}_i)$. Yeilding similar results to the optimal CI, FCI has become a popular alternative due to its computational simplicity [uses of fci]. For further reading, and uses of CI and FCI, see [ci refs].

2.2. Encryption Preliminaries

Used cryptographic notions and schemes are summarised here. In addition, the encoding of floating point numbers to integers suitable for encryption by the presented schemes is introduced as well.

2.2.1. Meeting Cryptographic Notions

Cryptographic notions are formal mathematical constructs used to prove the security of cryptographic schemes. A cryptographic notion applies to a type of cryptographic scheme, typically defined by a tuple of algorithms, *e.g.* ($\mathsf{Generate}, \mathsf{Encrypt}, \mathsf{Decrypt}$), and captures the capabilities of considered attackers and accepted leakage of information to these attackers, that is, what an attacker can do and what they can learn [katzIntroductionModernCryptography2008]. When a scheme of an appropriate form is proved to meet a cryptographic notion it is mathematically guaranteed that any attacker with the capabilities specified by the notion is limited in gaining information from encryptions as also specified by the notion.

Notion capabilities and leakages are dependent on the goal of a particular encryption scheme and are typically defined as a *cryptographic game* (or *experiment*) involving an adversary and the encryption scheme algorithms []. Proofs that schemes meet these notions are often based on existing proofs or assumptions believed to hold, and proved by contrapositive. Below, some cryptographic notions relevant to this thesis are introduced.

Indistinguishability under a Chosen Plaintext Attack (IND-CPA) This notion targets encryption schemes of the form ($\mathsf{Generate}, \mathsf{Encrypt}, \mathsf{Decrypt}$) and states that an attacker cannot distinguish between encryptions of unknown messages when they can encrypt chosen messages of their own. For detailed definitions and the formal cryptographic game, see [].

Aggregator Obliviousness (AO) The AO notion considers an environment of multiple participants where a single *aggregator* computes the summation of input from all other participants. An encryption scheme for this interaction is of the form ($\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{AggregateDecrypt}$). The notion states that no subset of colluding participants with access to all encryptions and the final summation can compute any more than a party with access to only the colluding participant inputs and the final summation can. The notion of AO and the relevant game are given can be found in [shiPrivacyPreservingAggregationTimeSeries2011].

Indistinguishability under an Ordered Chosen Plaintext Attack (IND-OCPA) IND-OCPA is the ideal notion of security for order-revealing encryption, stating that no attacker can distinguish between two sequences of encryptions when the numerical order of messages in the sequences is identical. The associated encryption scheme for the notion is of the form ($\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Compare}$). Additional details and the cryptographic game are given in [boldyreva order-preserving symmetric encryption].

As novel encryption schemes and cryptographic games are presented in this thesis, additional information on creating and proving cryptographic notions may be beneficial. For introductory methods and the structure of proofs, see [], while more advanced proofs can be found in [].

2.2.2. Paillier Homomorphic Encryption Scheme

The Paillier encryption scheme [paillierPublicKeyCryptosystemsBased1999] is an additively homomorphic encryption scheme that bases its security on the decisional composite residuosity assumption (DCRA) and meets the security notion of IND-CPA. Key generation of the Paillier scheme is performed by choosing two sufficiently large primes of an equal bit length, p and q, and computing $N = pq$ [katzIntroductionModernCryptography2008]. The public key is defined by N and the secret key by (p, q).

Encryption of a plaintext message $a \in \mathbb{Z}_N$, producing ciphertext $c \in \mathbb{Z}^*_{N^2}$, is computed by

$$c = (N+1)^a \rho^N \pmod{N^2} \tag{2.45}$$

for a randomly chosen $\rho \in \mathbb{Z}_N$. Here, ρ^N can be considered the noise term which hides the value $(N+1)^a \pmod{N^2}$, which due to the scheme construction, is an easily computable discrete logarithm. The decryption of a ciphertext is computed by

$$a = \frac{L(c^\lambda \pmod{N^2})}{L((N+1)^\lambda \pmod{N^2})} \pmod{N} \tag{2.46}$$

where $\lambda = \mathsf{lcm}(p-1, q-1)$ and $L(\psi) = \frac{\psi - 1}{N}$.

In addition to encryption and decryption, the following homomorphic functions are provided by the Paillier scheme. $\forall a_1, a_2 \in \mathbb{Z}_N$,

$$\mathcal{D}(\mathcal{E}(a_1)\mathcal{E}(a_2) \pmod{N^2}) = a_1 + a_2 \pmod{N}\,, \tag{2.47}$$

$$\mathcal{D}(\mathcal{E}(a_1)(N+1)^{a_2} \pmod{N^2}) = a_1 + a_2 \pmod{N}\,, \tag{2.48}$$

$$\mathcal{D}(\mathcal{E}(a_1)^{a_2} \pmod{N^2}) = a_1 a_2 \pmod{N}\,. \tag{2.49}$$

2.2.3. Joye-Libert Aggregation Scheme

The Joye-Libert privacy-preserving aggregation scheme [joyeScalableSchemePrivacyPreserving2013] is a scheme defined on time-series data and meets the security notion AO. Similarly to the Paillier scheme in section 2.2.2, it bases its security on the DCRA, however, an aggregation scheme generates private keys for individual participants and the aggregating party, requiring an additional trusted party to perform the private key generation and distribution.

Key generation is computed by choosing two equal-length and sufficiently large primes p and q, and computing $N = pq$. A hash function $H : \mathbb{Z} \to \mathbb{Z}^*_{N^2}$ is defined and the public key is set to (N, H). n private keys are generated by choosing sk_i, $i \in \{1, \dots, n\}$, uniformly from $\mathbb{Z}_{N^2}$ and distributing them to n participants (whose values are to be aggregated), while the last key is set as

$$sk_0 = -\sum_{i=1}^{n} sk_i\,, \tag{2.50}$$

and sent to the aggregator.

At any timestep k, a participant i can encrypt time-series plaintext data $a_i^{(k)} \in \mathbb{Z}_N$ to ciphertext $c_i^{(k)} \in \mathbb{Z}_{N^2}$ with

$$c_i^{(k)} = (N+1)^{a_i^{(k)}} H(k)^{sk_i} \pmod{N^2}\,. \tag{2.51}$$

Here, we can consider $H(k)^{sk_i}$ the noise term which hides the easily computable discrete logarithm $(N+1)^{a_i^{(k)}} \pmod{N^2}$.

When all encryptions $c_i^{(k)}$, $i \in \{1, \ldots, n\}$ are sent to the aggregator, summation and decryption of the aggregated sum are computed by the functions

$$c^{(k)} = H(k)^{sk_0} \prod_{i=1}^{n} c_i^{(k)} \pmod{N^2} \tag{2.52}$$

and

$$\sum_{i=1}^{n} a_i^{(k)} = \frac{c^{(k)} - 1}{N} \pmod{N}\,. \tag{2.53}$$

Correctness follows from $\sum_{i=0}^{n} sk_i = 0$, and thus

$$\begin{aligned}
&H(k)^{sk_0} \prod_{i=1}^{n} c_i^{(k)} \pmod{N^2} \\
\equiv &H(k)^{sk_0} \prod_{i=1}^{n} (N+1)^{a_i^{(k)}} H(k)^{sk_i} \pmod{N^2} \\
\equiv &H(k)^{\sum_{j=0}^{n} sk_j} \prod_{i=1}^{n} (N+1)^{a_i^{(k)}} \pmod{N^2} \\
\equiv &(N+1)^{\sum_{i=1}^{n} a_i^{(k)}} \pmod{N^2}\,,
\end{aligned}$$

removing all noise terms.

2.2.4. Lewi Order-Revealing Encryption Scheme

The Lewi order-revealing encryption scheme is a symmetric encryption scheme that allows for message values to be compared numerically after encryption. The scheme does not meet the ideal notion of security for order-revealing encryption, IND-OCPA, due to the inherent difficulty of this problem [chenettePracticalOrderRevealingEncryption2016] but rather meets a weaker simulation-based security given in [chenettePracticalOrderRevealingEncryption2016] that allows for some leakage.

To allow additional control over which encrypted values can be compared, the scheme provides two encryption functions, namely a *left* encryption and *right* encryption, such that comparisons can only take place between left and right encryptions. As complete equations for the scheme are lengthy and unnecessary for following this thesis, only notation will be introduced here. The two encryption equations provided can encrypt plaintexts $a_1, a_2 \in \mathbb{Z}$ with a secret key sk and functions

$$\mathcal{E}^{\mathsf{L}}_{sk}(a_1) \text{ and } \mathcal{E}^{\mathsf{R}}_{sk}(a_2) \tag{2.54}$$

denoting left and right encryption, respectively. Their comparison can be computed with a function

$$\mathcal{C}\left(\mathcal{E}^{\mathsf{L}}_{sk}(a_1), \mathcal{E}^{\mathsf{R}}_{sk}(a_2)\right) = \mathsf{cmp}(a_1, a_2)\,, \tag{2.55}$$

where

$$\mathsf{cmp}(a_1, a_2) = \begin{cases} -1 & a_1 < a_2 \\ 0 & a_1 = a_2 \\ 1 & a_1 > a_2 \end{cases}.$$

For details on implementation, see [lewi order revealing].

2.2.5. Encoding Numbers for Encryption

The Paillier encryption scheme and the Joye-Libert aggregation scheme in sections 2.2.2 and 2.2.3, respectively, both provide encryption on integer inputs in the modulo group $\mathbb{Z}_N$ given a large N. In addition, they provide homomorphic operations on these inputs after encryption, namely, the Paillier scheme provides addition and scalar multiplication with (2.47), (2.48) and (2.49) while the Joye-Libert scheme provides addition with (2.52). For this reason, real-valued estimation variables that may require encryption with these schemes, such as those introduced in section 2.1, require quantisation and integer mapping such that the operations are preserved after encryption. Throughout this thesis, we will rely on a generalised Q number encoding [oberstarFixedPointRepresentationFractional2007], applying scalar encoding and encryption operations elementwise on multidimensional data, due to applicability and implementation simplicity.

Quantisation to a subset of rational numbers is performed in terms of an output range $M \in \mathbb{N}$ and fractional precision $\phi \in \mathbb{N}$. This contrasts with the common definition in terms of total bits and fractional bits [oberstarFixedPointRepresentationFractional2007, schulzedarupEncryptedCooperativeControl2019, farokhiSecurePrivateControl2017] but allows for a direct mapping to integer ranges which are not a power of two, such as the group $\mathbb{Z}_N$. This rational subset, $\mathbb{Q}_{M,\phi}$, is defined by

$$\mathbb{Q}_{M,\phi} = \left\{ o \,\middle|\, \phi o \in \mathbb{N} \wedge -\left\lfloor \frac{M}{2} \right\rfloor \leq \phi o < \left\lfloor \frac{M}{2} \right\rfloor \right\}, \tag{2.56}$$

and can be used to quantise any real number $a \in \mathbb{R}$ by taking the nearest rational $o \in \mathbb{Q}_{M,\phi}$, that is, $\arg\min_{o \in \mathbb{Q}_{M,\phi}} |a - o|$. Mapping the rationals $\mathbb{Q}_{M,\phi}$, both positive and negative, to a group $\mathbb{Z}_M$ can then be achieved by modulo arithmetic. Additionally, we note that the Q number format requires a precision factor ϕ to be removed after each encoded multiplication. This is captured by a third parameter d; the number of additional precision factors present in encodings.

The function for *combined* quantisation and encoding, $\mathsf{E}_{M,\phi,d}(a)$, of a given number

$a \in \mathbb{R}$ and with output integer range $\mathbb{Z}_M$, precision ϕ and scaling for d prior encoded multiplications, is given by

$$\mathsf{E}_{M,\phi,d}(a) = \left\lfloor \phi^{d+1} a \right\rceil \pmod{M}. \tag{2.57}$$

Decoding of an integer $u \in \mathbb{Z}_M$ is given by

$$\mathsf{E}^{-1}_{M,\phi,d}(u) = \begin{cases} \dfrac{u \pmod{M}}{\phi^{d+1}}, & u \pmod{M} \leq \left\lfloor \dfrac{M}{2} \right\rfloor \\ -\dfrac{M - u \pmod{M}}{\phi^{d+1}}, & \text{otherwise} \end{cases}. \tag{2.58}$$

Encoding with (2.57) additionally provides two useful properties when used with homomorphic operations,

$$\mathsf{E}_{M,\phi,d}(a_1) + \mathsf{E}_{M,\phi,d}(a_2) \pmod{M} \approx \mathsf{E}_{M,\phi,d}(a_1 + a_2) \tag{2.59}$$

and

$$\mathsf{E}_{M,\phi,d}(a_1)\mathsf{E}_{M,\phi,d}(a_2) \pmod{M} \approx \mathsf{E}_{M,\phi,d+1}(a_1 a_2), \tag{2.60}$$

where deviation from equality stems from quantisation and its associated error

$$\left| \mathsf{E}^{-1}_{M,\phi,d}\left(\mathsf{E}_{M,\phi,d}(a)\right) - a \right| \leq \frac{1}{\phi}. \tag{2.61}$$

In general, choosing a large precision parameter ϕ reduces quantisation errors but risks overflow after too many multiplications. When the total number of encoded multiplications, d_{max}, and the largest value to be encoded, a_{max}, are known, ϕ can be chosen to avoid overflow by satisfying

$$\left| \phi^{d_{\mathsf{max}}+1} a_{\mathsf{max}} \right| < \left\lfloor \frac{M}{2} \right\rfloor. \tag{2.62}$$

In practice, we set $M = N$ to use the Paillier or Joye-Libert schemes, such that the properties (2.59) and (2.60) can be used with the homomorphic operations of the schemes. Equation (2.62) can then be ignored as N is typically very large ($N > 2^{1024}$) and ϕ can be made sufficiently large to make quantisation errors negligible.

3. Estimate Fusion on an Untrusted Cloud

3.1. Problem Formulation

The details of the confidential estimate fusion problem that we consider are In this work, we consider an arbitrary time-varying process defined by its state $\underline{x}_k \in \mathbb{R}^n$ for all timesteps $k \in \mathbb{N}$. This process is estimated by m individual estimators i, $1 \leq i \leq m$, each producing a state estimate and an estimate error covariance,

$$\hat{\underline{x}}_{k,i} \in \mathbb{R}^n \text{ and } \mathbf{P}_{k,i} \in \mathbb{R}^{n \times n}\,, \tag{3.1}$$

respectively, at every timestep k. Our aim at every timestep is for all estimates $\hat{\underline{x}}_{k,i}$ and errors $\mathbf{P}_{k,i}$, $1 \leq i \leq m$, to be sent to a cloud server where a fused estimate and error covariance,

$$\hat{\underline{x}}_{k,\mathsf{fus}} \in \mathbb{R}^n \text{ and } \mathbf{P}_{k,\mathsf{fus}} \in \mathbb{R}^{n \times n}\,, \tag{3.2}$$

respectively, are computed and are consistent with the estimates in (3.1).

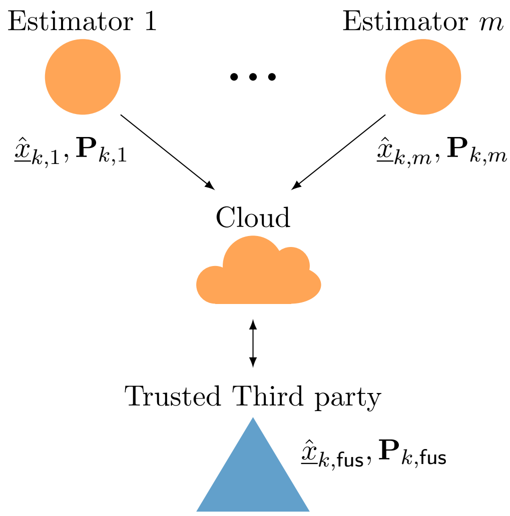
Simultaneously, we consider the desired security of the fusion process. The fusing cloud and estimators are treated as *honest-but-curious*, that is, they follow protocols correctly but may use learned information for malicious gain, and a trusted third party exists that can query the cloud at any timestep k to obtain the fusion results in (3.2). We aim for the cloud, estimators and eavesdroppers to learn no additional information from observed estimates and fusions, (3.1) and (3.2), respectively, beyond their local estimates. To guarantee this, cryptographic *Indistinguishability under the Chosen Plaintext Attack* (IND-CPA) [katzIntroductionModernCryptography2008] is desired for all transmitted and processed information. This is in line with common security goals in the field and is suitable for homomorphic encryption. The communications of this scenario are summarised graphically in figure [fig:layout].

3.2. Confidential Cloud Fusion Leaking Fusion Weights

3.2.1. Secure Fast Covariance Intersection

Two-sensor Case

In this section, we will introduce the Secure FCI (SecFCI) fusion algorithm for the two-sensor case, before extending it to the n sensor case in section [sec:multi_secfci]. The network model we consider is described in section [subsec:problem_formulation], where sensors are capable of running local estimators, as well as the PHE and ORE encryption schemes from section [sec:encryption]. Each sensor i computes its state estimate $\hat{\underline{x}}_i$ and



covariance matrix $\mathbf{P}_i$ and sends relevant encrypted information to an untrusted cloud fusion center. The querying party is the key holding party and generates the PHE public key pk, secret key sk, and ORE symmetric key k. pk is made available to all parties in the network, and k is made available to the sensors only, via any standard public-key scheme such as RSA [rivestMethodObtainingDigital1978]. When encrypting with ORE key k, individual sensors are limited to using only L or R ORE encryption to reduce local information leakage. Thus, consecutive ORE encryptions from any sensor cannot be used to infer local information directly, and can only be compared to encryptions from sensors using the alternate ORE encryption.

From [eqn:ci_cov_estimate], we can see that both CI fusion equations can be computed on PHE encryptions of sensor information vectors and information matrices, given valid unencrypted values for each ω_i. For this reason, we allow the leakage of all weights ω_i. Thus, in the two-sensor case, homomorphic fusion is computed by

$$\mathcal{E}(\mathbf{P}^{-1}) = \mathcal{E}(\mathbf{P}_1^{-1})^{\omega_1}\mathcal{E}(\mathbf{P}_2^{-1})^{(1-\omega_1)} \tag{3.3}$$

and

$$\mathcal{E}(\mathbf{P}^{-1}\hat{\underline{x}}) = \mathcal{E}(\mathbf{P}_1^{-1}\hat{\underline{x}}_1)^{\omega_1}\mathcal{E}(\mathbf{P}_2^{-1}\hat{\underline{x}}_2)^{(1-\omega_1)} \ , \tag{3.4}$$

where we note that $\omega_2 = 1-\omega_1$ due to the CI requirement [eqn:ci_omega_sum_bound]. We also note that in [eqn:paillier_ci_cov] and [eqn:paillier_ci_estimate], each resulting value will have exactly one encoding multiplication factor to remove, and can be decoded exactly by using [eqn:qmn_mult_decode].

All that remains for computing CI homomorphically, in the two-sensor case, is the calculation of parameter ω_1. For this, we approximate the solution to FCI. Since our encoding scheme in section [subsec:encoding] does not allow division, the exact result of [eqn:fci_2sen_omega_sum_eq] is approximated. This is accomplished by discretizing ω_i by step-size s, such that $s < 1$ and $p = 1/s \in \mathbb{Z}$, and approximating [eqn:fci_2sen_omega_sum_eq] with ORE. An ordered discretization of values $\omega^{(x)}$ is de-

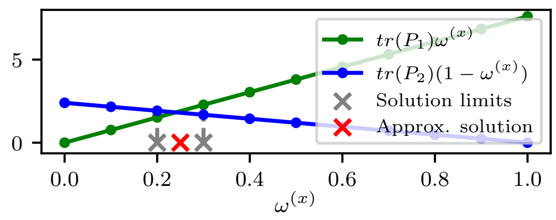


Figure 3.1.: Approximation of ω_1 with discretization step-size $s = 0.1$. Only comparisons between line points are used.

fined by

$$[\omega^{(1)}, \ldots, \omega^{(p)}] = [0, s, \ldots, 1-s, 1] \ , \tag{3.5}$$

and computed by each sensor i. Each $\omega^{(x)}$ is multiplied by $\mathrm{tr}(\mathbf{P}_i)$ and encrypted with ORE key k. Sensor 1's list is defined by

$$[\mathcal{E}^{L}_{ORE}(\omega^{(1)}\,\mathrm{tr}(\mathbf{P}_1)), \ldots, \mathcal{E}^{L}_{ORE}(\omega^{(p)}\,\mathrm{tr}(\mathbf{P}_1))] \ , \tag{3.6}$$

and similarly sensor 2's by

$$[\mathcal{E}^{R}_{ORE}(\omega^{(1)}\,\mathrm{tr}(\mathbf{P}_2)), \ldots, \mathcal{E}^{R}_{ORE}(\omega^{(p)}\,\mathrm{tr}(\mathbf{P}_2))] \ . \tag{3.7}$$

Note that Sensor 1 uses only L ORE while sensor 2 uses only R ORE, and that both lists are ordered. Lists (3.6) and (3.7) are sent alongside PHE encryptions of local information vector and information matrix estimates to the fusion center which uses them to estimate the FCI values of ω_1 and ω_2.

From [eqn:fci_2sen_omega_sum_eq] we know that ω_1 must satisfy

$$\omega_1\,\mathrm{tr}(\mathbf{P}_1) = (1-\omega_1)\,\mathrm{tr}(\mathbf{P}_2) \ . \tag{3.8}$$

If we reverse (3.7), we obtain a list equivalent to one with values $\mathcal{E}^{R}_{ORE}((1-\omega^{(x)})\,\mathrm{tr}(\mathbf{P}_2))$ for each discretization step x. When the reversed list is decrypted and plotted over (3.6) the intersection gives the solution to (3.8) and therefore, (**??**). However, (3.6) and reversed (3.7) consist of L and R ORE encryptions respectively, and the intersection must be approximated by locating consecutive $\omega^{(x)}$ discretizations where the sign of comparisons changes. This can be seen in Fig. 3.1, and can be performed in $O(\log p)$ ORE comparisons using a binary search. Consecutive $\omega^{(x)}$ and $\omega^{(x+1)}$ for which list comparisons differ can be used to estimate the true intersection, and ω_1, by

$$\omega_1 \approx 0.5(\omega^{(x)} + \omega^{(x+1)}) \ . \tag{3.9}$$

In the case a comparison returns equality, the exact value of $\omega^{(x)}$ can be taken to be ω_1.

The fusion center can then use its values for ω_1 and $\omega_2 = 1-\omega_1$ and the received PHE encryptions of local information vectors and information matrices to compute (3.3) and (3.4).

Multi-sensor Case

When computing the SecFCI fusion for n sensors, we solve (**??**) homomorphically by computing

$$\mathcal{E}(\mathbf{P}^{-1}) = \mathcal{E}(\mathbf{P}_1^{-1})^{\omega_1} \cdots \mathcal{E}(\mathbf{P}_n^{-1})^{\omega_n} \tag{3.10}$$

and

$$\mathcal{E}(\mathbf{P}^{-1}\underline{\hat{x}}) = \mathcal{E}(\mathbf{P}_1^{-1}\underline{\hat{x}}_1)^{\omega_1} \cdots \mathcal{E}(\mathbf{P}_n^{-1}\underline{\hat{x}}_n)^{\omega_n} \ . \tag{3.11}$$

As with the two-sensor case, encoded results from (3.10) and (3.11) contain exactly one multiplication factor to remove and can be decoded exactly with [eqn:qmn_mult_decode]. Again we are just left with the task of computing the plaintext weights $\omega_1, \ldots, \omega_n$.

Our approach to the n sensor case is to solve all $n-1$ conditions in (**??**) using the two-sensor method, and combining partial solutions to compute the final result. When we consider a Euclidean dimension for each ω_i, partial solutions can be considered geometrically as hyperplanes of $n-2$ dimension, over the $n-1$ dimensional solution space given by (**??**).

This can be visualized in the three sensor case, which requires solving partial solutions

$$\omega_1 \operatorname{tr}(\mathbf{P}_1) - \omega_2 \operatorname{tr}(\mathbf{P}_2) = 0,\ \omega_1 + \omega_2 = 1 - \omega_3 \tag{3.12}$$

and

$$\omega_2 \operatorname{tr}(\mathbf{P}_2) - \omega_3 \operatorname{tr}(\mathbf{P}_3) = 0,\ \omega_2 + \omega_3 = 1 - \omega_1 \ . \tag{3.13}$$

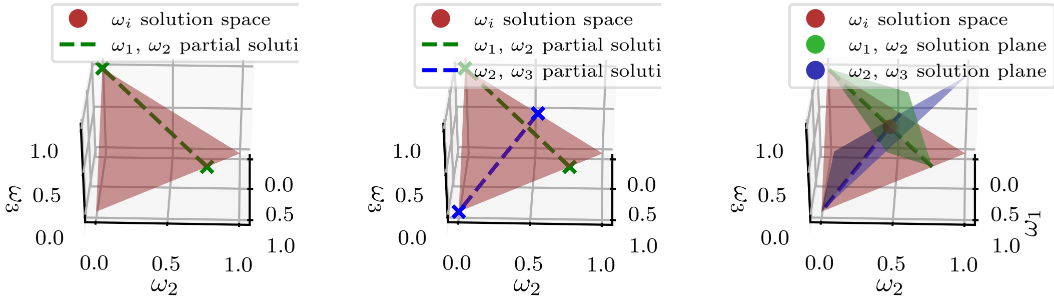
We can use the two-sensor method from section 3.2.1 to solve (3.12) exactly when $\omega_3 = 0$, and know that when $\omega_3 = 1$, then $\omega_1 = \omega_2 = 0$. These two points are enough to define the two-dimensional partial solution (3.12) which can be seen plotted over the possible solution space in Fig. 3.2(a). Fig. 3.2(b) shows both partial solutions (3.12) and (3.13) plotted over the solution space. The final solution from all partial solutions is computed by finding their intersection. This can be seen in Fig. 3.2(b) as the intersection of the (ω_1, ω_2) and (ω_2, ω_3) partial solution lines.

To simplify computing the partial solution intersection, we define equivalent planes for each of the partial solutions, perpendicular to the solution space, in the form

$$a_1\omega_1 + a_2\omega_2 + a_3\omega_3 + a_4 = 0 \ , \tag{3.14}$$

and solve the resulting linear system for finding the intersection of all planes and the solution space. This is given by

$$\begin{bmatrix} a_1^{(1)} & a_2^{(1)} & a_3^{(1)} \\ a_1^{(2)} & a_2^{(2)} & a_3^{(2)} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} a_3^{(1)} \\ a_4^{(2)} \\ 1 \end{bmatrix} \ , \tag{3.15}$$



(a) Partial solution to (3.12). (b) Partial solutions to (3.12) and (3.13). (c) Partial solutions as planes.

Figure 3.2.: Partial solutions over ω_1, ω_2, and ω_3 solution space.

where $a_i^{(j)}$ denotes parameter i of partial solution j, and has been shown visually in Fig. 3.2(c).

In the n sensor case, we can similarly solve partial solutions by first using the method from section 3.2.1 to solve equations with two parameters ω_k and ω_{k+1} when letting all $\omega_i = 0,\ i \neq k, k+1$. For each equation we can then compute remaining partial solution points at $\omega_i = 1,\ i \neq k, k+1$ with $\omega_j = 0,\ j \neq i$. Perpendicular hyperplanes can then be similarly defined in the form

$$a_1\omega_1 + \cdots + a_n\omega_n + a_{n+1} = 0\ . \tag{3.16}$$

Due to their inherent orthogonality, and that all meaningful covariance traces are strictly positive, the $n-1$ partial solution hyperplanes are guaranteed to intersect at exactly one point. The hyperplane intersection results in the linear system

$$\begin{bmatrix} a_1^{(1)} & a_2^{(1)} & \cdots & a_n^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ a_1^{(n-1)} & a_2^{(n-1)} & \cdots & a_n^{(n-1)} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_{n-1} \\ \omega_n \end{bmatrix} = \begin{bmatrix} a_{n+1}^{(1)} \\ \vdots \\ a_{n+1}^{(n-1)} \\ 1 \end{bmatrix}\ , \tag{3.17}$$

and gives the solution to the SecFCI ω_i weights.

As all $O(n \log p)$ ORE comparisons are done between sequential sensors i and $i+1$, L and R ORE encryptions can be used to the same effect as for the two-sensor case. The ORE ordered list sent from each sensor i is given by

$$\begin{aligned} &[\mathcal{E}^L_{ORE}(\omega^{(1)}\operatorname{tr}(\mathbf{P}_i)), \dots, \mathcal{E}^L_{ORE}(\omega^{(p)}\operatorname{tr}(\mathbf{P}_i))],\ i \text{ odd} \\ &[\mathcal{E}^R_{ORE}(\omega^{(1)}\operatorname{tr}(\mathbf{P}_i)), \dots, \mathcal{E}^R_{ORE}(\omega^{(p)}\operatorname{tr}(\mathbf{P}_i))],\ i \text{ even.} \end{aligned} \tag{3.18}$$

When combining (3.18) with PHE encryptions of local information vectors and informa-

Table 3.1.: Computation complexity of encryption operations.

Operation	**Complexity**
Paillier enc.	$O(\log^3 N)$
Paillier dec.	$O(\log^3 N)$
Paillier add.	$O(\log^2 N)$
Paillier scalar mult.	$O(\log^3 N)$
Lewi L enc.	$O(\log^2 N)$
Lewi R enc.	$O(\log^2 N)$
Lewi comp.	$O(\log^2 N)$

tion matrices, SecFCI can be computed entirely homomorphically by (3.10) and (3.11).

Briefly considering the security of our scheme, we note that any leaked information from ORE lists (3.18), as described in [chenettePracticalOrderRevealingEncryption2016], can be considered a subset of knowing the estimated fusion weights $\omega_1, \dots, \omega_n$, which specify relative sizes of sensor covariance traces, and we already consider public. Thus only IND-CPA and IND-OCPA (after accounting for leakage through public weights) encryptions are made available to the fusion centre.

3.2.2. Computational Complexity

Given the state estimates and estimate errors at each sensor, we wish to show the computational complexity of the SecFCI algorithm for the n sensor case. We will assume that both Lewi ORE and Paillier PHE schemes use the same length security parameter (and equivalently key size), such that $\lambda_{Lewi} = \lambda_{Paillier} = \log N$, where λ_s represents encryption scheme s's security parameter, and N the Paillier modulus and encryptable integer limit. We also note the distinction between floating-point or small integer operations, which are typically treated as having $O(1)$ runtime, and large integer operations whose complexities are dependent on bit length. While architectures exist for speeding up encryption operations [gueronIntelAdvancedEncryption2010], we consider software implementations and treat large integer operations in terms of bit operations explicitly.

From [paillierPublicKeyCryptosystemsBased1999,lewiOrderRevealingEncryptionNew2016], and the assumptions made above, we have summarized the operation complexities of the two schemes in Table 3.1. In contrast to some current FHE schemes, these operations are of a much lower complexity than [vandijkFullyHomomorphicEncryption2010a], which has complexity $O(\lambda^{10})$ for integer operations, and [stehleFasterFullyHomomorphic2010], which computes single bit operations in $O(\lambda^{3.5})$ adding significant overhead for integer arithmetic.

Finally, applying the operations from Table 3.1 to the SecFCI algorithm, we summarize the total complexity of SecFCI at the sensors and the fusion centre in Table 3.2, with the unencrypted complexities of FCI shown for reference.

Table 3.2.: Computation complexity at sensors and fusion centre.

	FCI	SecFCI
Sensors	$O(1)$	$O\left(p\log^2 N + \log^3 N\right)$
Fusion	$O(n^3)$	$O\left(n\log p \log^2 N + n\log^3 N + n^3\right)$

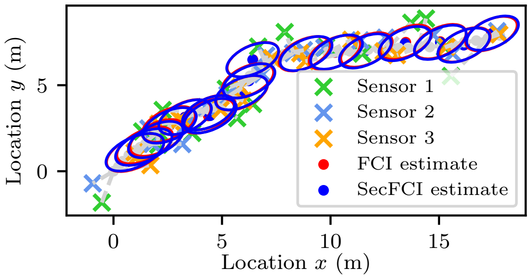


Figure 3.3.: Tracking simulation comparing SecFCI and FCI.

3.2.3. Security Analysis

3.2.4. Simulation

We have implemented a simulation to demonstrate the accuracy of SecFCI approximating FCI. Three sensors independently measure a constant-speed linear process and simultaneously run a Kalman filter on their measurements. Estimates are sent both encrypted and unencrypted to a fusion centre that computes the SecFCI and FCI fusions on the received data respectively. Encrypted estimates are comprised of PHE encryptions of the information vector and information matrix, $\mathcal{E}(\mathbf{P}_i^{-1}\underline{\hat{x}}_i)$ and $\mathcal{E}(\mathbf{P}_i^{-1})$, in addition to the ORE list given by (3.18) with discretization step $s = 0.1$. Unencrypted estimates consist of the state estimate $\underline{\hat{x}}_i$ and covariance $\mathbf{P}_i$. The trajectory and fused estimates are shown in Fig. 3.3.

To derive an upper bound on the accuracy difference between SecFCI and FCI, we note the two factors which introduce inconsistency between the two methods: the encoding method from section 3.2.2, and the difference in fusion weights. Due to the possibility of choosing sufficiently large integer and fractional bit lengths i and f, we will only consider the error caused by the difference in weights. We will treat this error as the distance between respective weight vectors

$$\begin{aligned}\underline{\omega}_{SecFCI} &= (\omega_{1,SecFCI}, \dots, \omega_{n,SecFCI}) \\ \underline{\omega}_{FCI} &= (\omega_{1,FCI}, \dots, \omega_{n,FCI}) \ ,\end{aligned} \tag{3.19}$$

where $\omega_{i,s}$ denotes weight ω_i from algorithm s. From section 3.2.1 we see that the largest difference $|\omega_{i,FCI} - \omega_{i,SecFCI}|$ is strictly bounded by $s/2$. As shown in section 3.2.1, when more sensors are involved, a tighter bound on this difference is dependent on the value

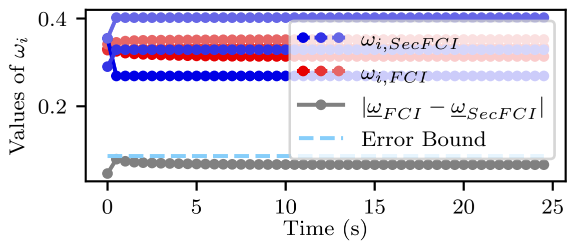


Figure 3.4.: $\underline{\omega}_{SecFCI}$ and $\underline{\omega}_{FCI}$ components.

of $\underline{\omega}_{i,FCI}$, but will remain strictly bounded by $s/2$. Therefore, we can give a strict upper bound on the distance between weight vectors as

$$|\underline{\omega}_{FCI} - \underline{\omega}_{SecFCI}| < 0.5\sqrt{ns^2} \ . \tag{3.20}$$

Finally, components of $\omega_{i,SecFCI}$, $\omega_{i,FCI}$ and the errors $|\underline{\omega}_{FCI} - \underline{\omega}_{SecFCI}|$, have been plotted over time in Fig. 3.4, and show the computed error bound when $n = 3$ and $s = 0.1$.

3.3. Confidential Cloud Fusion Without Leaking Fusion Weights

With the problem and preliminaries introduced, we can now present our encrypted FCI method that leaks no estimator information to the fusing cloud. The core idea behind the method is to postpone the evaluation of operations that cannot be performed homomorphically until partial results are queried and decrypted by the key-holding third party. The remaining operations can then be evaluated on unencrypted inputs to produce the correct results.

First, we note that the FCI fusion equations [eqn:ci_cov_fusion] and [eqn:ci_est_fusion] can be rearranged and substituted with weights [eqn:fci_weights] to obtain the equations

$$\mathbf{P}_{k,\mathsf{fus}} = \left(\left(\sum_{i=1}^{m} \frac{1}{\mathrm{tr}(\mathbf{P}_{k,i})} \right)^{-1} \sum_{i=1}^{m} \frac{1}{\mathrm{tr}(\mathbf{P}_{k,i})} \mathbf{P}_{k,i}^{-1} \right)^{-1} \tag{3.21}$$

and

$$\underline{\hat{x}}_{k,\mathsf{fus}} = \mathbf{P}_{k,\mathsf{fus}} \left(\sum_{i=1}^{m} \frac{1}{\mathrm{tr}(\mathbf{P}_{k,i})} \right)^{-1} \sum_{i=1}^{m} \frac{1}{\mathrm{tr}(\mathbf{P}_{k,i})} \mathbf{P}_{k,i}^{-1} \underline{\hat{x}}_{k,i} \, . \tag{3.22}$$

In this form, innermost summations

$$\sum_{i=1}^{m} \frac{1}{\operatorname{tr}(\mathbf{P}_{k,i})}, \sum_{i=1}^{m} \frac{1}{\operatorname{tr}(\mathbf{P}_{k,i})} \mathbf{P}_{k,i}^{-1} \text{ and } \sum_{i=1}^{m} \frac{1}{\operatorname{tr}(\mathbf{P}_{k,i})} \mathbf{P}_{k,i}^{-1} \underline{\hat{x}}_{k,i} \tag{3.23}$$

combine information from individual estimators i and are computable homomorphically given suitable encryptions. Encryptions of these sums can then be decrypted by the key-holding third party, before remaining inversions and multiplications in (3.21) and (3.22) can be computed to obtain the final results. To depict this process, pseudocode for the encryption at estimators, fusion at the cloud and decryption by the third party are provided in algorithms 1, 2 and 3, respectively.

Algorithm 1 Estimator Encryption

procedure ESTIMATE(i, k, pk, ϕ)
  Estimate $\underline{\hat{x}}_{k,i}$ locally
  Estimate $\mathbf{P}_{k,i}$ locally
  ▷ Public key is encoding and encryption modulus
  $N \leftarrow \mathsf{pk}$
  ▷ Encode scaling, covariance and estimate components
  $\tilde{s}_{k,i} \leftarrow \mathsf{E}_{N,\phi}\left(\frac{1}{\operatorname{tr}(\mathbf{P}_{k,i})}\right)$
  $\tilde{\mathbf{C}}_{k,i} \leftarrow \mathsf{E}_{N,\phi}\left(\frac{1}{\operatorname{tr}(\mathbf{P}_{k,i})}\mathbf{P}_{k,i}^{-1}\right)$
  $\underline{\tilde{e}}_{k,i} \leftarrow \mathsf{E}_{N,\phi}\left(\frac{1}{\operatorname{tr}(\mathbf{P}_{k,i})}\mathbf{P}_{k,i}^{-1}\underline{\hat{x}}_{k,i}\right)$
  ▷ Encrypt scaling, covariance and estimate components
  $s_{k,i} \leftarrow \mathcal{E}_{\mathsf{pk}}\left(\tilde{s}_{k,i}\right)$
  $\mathbf{C}_{k,i} \leftarrow \mathcal{E}_{\mathsf{pk}}\left(\tilde{\mathbf{C}}_{k,i}\right)$
  $\underline{e}_{k,i} \leftarrow \mathcal{E}_{\mathsf{pk}}\left(\underline{\tilde{e}}_{k,i}\right)$
  Send $s_{k,i}$, $\mathbf{C}_{k,i}$ and $\underline{e}_{k,i}$ to fusing cloud
end procedure

Remark. Along with allowing the summations to be performed homomorphically on the cloud, we note that this form of the FCI also allows the cloud's partial fusion operations to be evaluated sequentially. This can be seen in algorithm 2, where individual components $s_{k,i}$, $\mathbf{C}_{k,i}$ and $\underline{e}_{k,i}$ from each estimator can continue to be aggregated as additional estimators send their estimate information. This, in turn, supports the dynamic joining and leaving of estimators in the network without affecting the cloud or the operations

Algorithm 2 Cloud Fusion

procedure $\textsc{Fuse}(k, \mathsf{pk})$
    Receive $s_{k,i}$, $\mathbf{C}_{k,i}$ and $\underline{e}_{k,i}$ for all $1 \leq i \leq m$
    $N \leftarrow \mathsf{pk}$
    $s_k \leftarrow \prod_{i=1}^{m} s_{k,i} \pmod{N^2}$
    $\mathbf{C}_k \leftarrow \otimes_{i=1}^{m} \mathbf{C}_{k,i} \pmod{N^2}$
    $\underline{e}_k \leftarrow \otimes_{i=1}^{m} \underline{e}_{k,i} \pmod{N^2}$
    Store s_k, $\mathbf{C}_k$ and $\underline{e}_k$ in case of query
end procedure

Algorithm 3 Fusion Query

procedure $\textsc{GetResult}(k, \mathsf{pk}, \mathsf{sk}, \phi)$
    Query and receive s_k, $\mathbf{C}_k$ and $\underline{e}_k$ from fusing cloud
    $N \leftarrow \mathsf{pk}$
    ▷ Decrypt
    $\tilde{s}_k \leftarrow \mathcal{D}_{\mathsf{sk}}(s_k)$
    $\tilde{\mathbf{C}}_k \leftarrow \mathcal{D}_{\mathsf{sk}}(\mathbf{C}_k)$
    $\tilde{\underline{e}}_k \leftarrow \mathcal{D}_{\mathsf{sk}}(\underline{e}_k)$
    ▷ Decode
    $\bar{s}_k \leftarrow \mathsf{E}_{N,\phi}^{-1}(\tilde{s}_k)$
    $\bar{\mathbf{C}}_k \leftarrow \mathsf{E}_{N,\phi}^{-1}\left(\tilde{\mathbf{C}}_k\right)$
    $\bar{\underline{e}}_k \leftarrow \mathsf{E}_{N,\phi}^{-1}(\tilde{\underline{e}}_k)$
    ▷ Compute Fusion
    $\mathbf{P}_{k,\mathsf{fus}} \leftarrow \left(\bar{s}_k^{-1} \cdot \bar{\mathbf{C}}_k\right)^{-1}$
    $\hat{\underline{x}}_{k,\mathsf{fus}} \leftarrow \mathbf{P}_{k,\mathsf{fus}} \cdot \bar{s}_k^{-1} \cdot \bar{\underline{e}}_k$
    return $\hat{\underline{x}}_{k,\mathsf{fus}}$, $\mathbf{P}_{k,\mathsf{fus}}$
end procedure

Table 3.3.: Computation complexity of encryption operations.

Operation	**Complexity**
Encryption	$O(\log^3 N)$
Decryption	$O(\log^3 N)$
Addition	$O(\log^2 N)$
Scalar mult.	$O(\log^3 N)$

Table 3.4.: Computation complexity at parties during fusion.

	FCI	**Our Method**
Estimator	$O(1)$	$O\left(n^2 \log^3 N\right)$
Fusion	$O(mn^3)$	$O\left(mn^2 \log^2 N\right)$
Third party	$O(1)$	$O\left(n^2 \log^3 N + n^3\right)$

of a trusted third party. The security implications of such an extension are discussed further in section 3.3.2.

3.3.1. Computational Complexity

The method described provides a level of security when relying on an untrusted cloud for fusing estimates. However, the added reliance on an encryption scheme and the additional computations at the third party intuitively increase the computational complexity of the algorithm and the required capabilities of participating parties. Here, we present the complexity of operations during fusion, required by each party at every timestep k. We assume encoding and decoding operations have complexity $O(1)$ (due to their comparative insignificance when compared to associated encryption and decryption operations) and use [12] to obtain the encryption complexities of the Paillier encryption scheme in table 3.3, with security parameter $\lambda = \log N$. In table 3.4, we compare the complexities of the unencrypted FCI algorithm and the method presented in this work. It can be seen that the burden of computation is greatly increased at the estimators and the third party, in particular when dimension n is large and when a long encryption key N is used. Naturally, an application of the proposed method would need to consider these requirements in terms of computation time and required hardware.

3.3.2. Security Analysis

The provable security of the presented method is relatively straightforward. Our aim for IND-CPA security of all information received by the cloud, sent by the estimators or observable by eavesdroppers is achieved by the homomorphic Paillier encryption scheme. Since all transmitted information is encrypted and the cloud, estimators and eavesdroppers do not hold the secret key sk, IND-CPA is met at all parties.

We note, however, an implicit assumption made when encrypting multidimensional data element-wise. While individual elements are indistinguishable, element-wise encryption does not encrypt the estimate's dimension n, which remains implicitly public. While existing methods allow the complete homomorphic encryption of vectors [alexandruPrivateWeightedSum2020], they are left for future work and considered beyond the scope of this work. Instead, we acknowledge the implicit leakage of n and note that, while this may leak information about the fusion's use case, state estimates remain hidden. Additionally, intuitive extensions to the scheme, such as the dynamic joining and leaving of estimators in remark 3.3, may introduce further implicit leakages that must be considered if security is analysed. In this example, the periodic estimation may leak to the cloud when estimators are within an estimation range or context, and a solution may be sending dummy measurements with $s_{k,i} = \mathcal{E}_{\mathsf{pk}}(\mathsf{E}_{N,\phi}(0))$ when estimator i is out of range. This extension is presented only as an example of when care needs to be taken to maintain desired security goals, but in general, extensions and solutions are task-dependent and not a focus of this work.

3.3.3. Simulation

Fusion estimates and error covariances from the proposed encrypted FCI method differ from unencrypted FCI only when quantisation errors are large or summation overflows occur. As stated in section [subsec:encoding], when the Paillier modulus N is large, these errors can often be considered negligible. In this section, we demonstrate this similarity in performance between the encrypted and unencrypted FCI fusion algorithms with a simulation. Code was written in the Python programming language, using the `phe` Paillier encryption scheme library [PythonPaillier2013] and a 512 bit length key (bit length of N). The simulation implements a linear constant velocity model,

$$\underline{x}_k = \begin{bmatrix} 1 & 0.5 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \underline{x}_{k-1} + \underline{w}_k \,, \tag{3.24}$$

with noise term $\underline{w}_k \sim \mathcal{N}(\underline{0}, \mathbf{Q})$ and

$$\mathbf{Q} = 10^{-3} \cdot \begin{bmatrix} 0.42 & 1.25 & 0 & 0 \\ 1.25 & 5 & 0 & 0 \\ 0 & 0 & 0.42 & 1.25 \\ 0 & 0 & 1.25 & 5 \end{bmatrix} . \tag{3.25}$$

At each timestep k, the system state $\underline{x}_k$ is estimated by $m = 4$ estimators, $1 \leq i \leq 4$, using a standard linear Kalman filter (KF) [haugBayesianEstimationTracking2012] and producing estimates and error covariances $\hat{\underline{x}}_{k,i}$ and $\mathbf{P}_{k,i}$, respectively. The measurements used by the KF, $\underline{z}_{k,i}$, follow the measurement models

$$\underline{z}_{k,i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \underline{x}_k + \underline{v}_{k,i} \,, \tag{3.26}$$

with noise terms $\underline{v}_{k,i} \sim \mathcal{N}(\underline{0}, \mathbf{R}_i)$ and covariances sampled indepedently, resulting in

$$\begin{aligned} \mathbf{R}_1 &= \begin{bmatrix} 4.77 & -0.15 \\ -0.15 & 4.94 \end{bmatrix}, \ \mathbf{R}_2 = \begin{bmatrix} 2.99 & -0.55 \\ -0.55 & 4.44 \end{bmatrix}, \\ \mathbf{R}_3 &= \begin{bmatrix} 2.06 & 0.68 \\ 0.68 & 1.96 \end{bmatrix} \text{ and } \mathbf{R}_4 = \begin{bmatrix} 1.17 & 0.80 \\ 0.80 & 0.64 \end{bmatrix}. \end{aligned} \tag{3.27}$$

The fusion results of 1000 simulation runs are shown in figure 3.5. From the figure, we

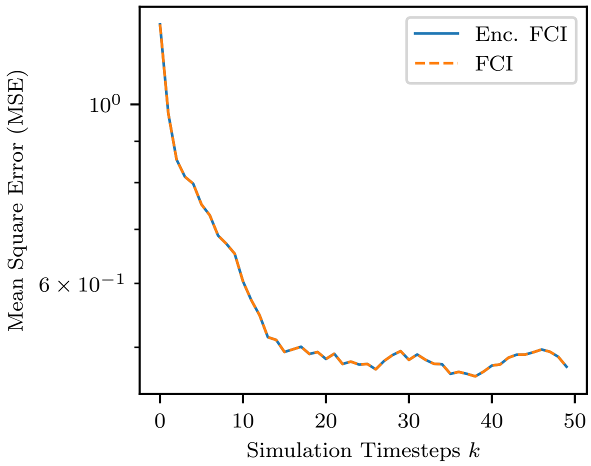


Figure 3.5.: Average RMSE of encrypted and unencrypted FCI fusion over 1000 simulations.

can see the expected similarity in performance between the encrypted and unencrypted FCI methods. Additionally, we note that the current recommended key length for the Paillier encryption scheme is 2048 bits [barkerRecommendationPairWiseKey2019], easily supporting a modulus N and fractional precision ϕ that guarantee similar performance.

3.4. Conclusions on Confidential Estimate Fusion

4. Distributed Non-Linear Measurement Fusion with Untrusted Participants

4.1. Problem Formulation

In this work, we consider the context of privacy-preserving range sensor navigation, where we want no sensor to learn any information about the navigator or other sensors beyond their local measurements, and the navigator not to learn any information about individual sensors beyond its location estimate. The problem is two-fold, in that we require explicit cryptographic requirements with a suitable encryption scheme meeting them as well as an estimation scheme that can use the encryption in the context of range-only navigation.

To give a formal cryptographic requirement in a distributed setting, we must first consider the communication requirements of our context and define attacker capabilities and the desired security of a suitable encryption scheme. In this section, we will define a communication protocol and the relevant formal definition of security we aim to achieve, followed by the estimation problem to which we will apply it.

4.1.1. Formal Cryptographic Problem

The communication between the navigator and sensors in our estimation problem will be decomposed into a simple two-step bi-directional protocol that will simplify defining formal security. In section 4.2.2, we will show how this protocol is sufficient to compute the location estimate at a navigator while meeting our desired privacy goals. The communication protocol is as follows.

At every *instance* t (used to distinguish from an estimation *timestep*), the navigator first broadcasts m weights $\omega_j^{(t)}, j \in \{1, \dots, m\}$ to all sensors $i \in \{1, \dots, n\}$, who individually compute linear combinations $l_i^{(t)} = \sum_{j=1}^{m} a_{j,i}^{(t)} \omega_i^{(t)}$ based on their measurement data $a_{j,i}$. Linear combinations are then sent back to the navigator, who computes their sum $\sum_{i=1}^{n} l_i^{(t)}$. This two-step linear combination aggregation protocol has been visually displayed in figure 4.1. In addition, we note that an alternative approach to the two-step protocol is computing $\sum_{j=1}^{m}(\omega_j^{(t)} \sum_{i=1}^{n} a_{i,j}^{(t)})$ at the navigator, requiring only values $a_{i,j}^{(t)}, j \in \{1, \dots, m\}$ to be sent from each sensor i. We justify the use of bi-directional communication by reducing communication costs when the number of weights is larger than the number of sensors, $m > n$, and by sending fewer weights in the presence of repeats, as will be shown to be the case in section 4.2.2.

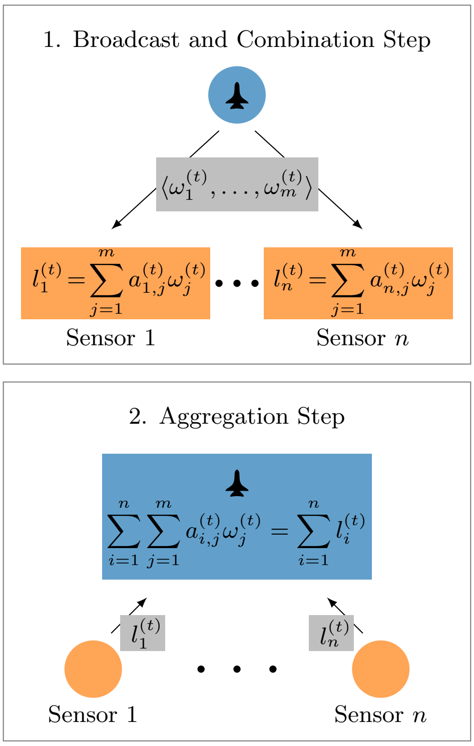


Figure 4.1.: Required linear combination aggregation steps at instance t.

Before giving a formal definition for the construction and security of our desired encryption scheme, we make the following assumptions about the capabilities of the participants.

Global Navigator Broadcast We assume that broadcast information from the navigator is received by *all* sensors involved in the protocol.

Consistent Navigator Broadcast We assume that broadcast information from the navigator is received equally by all sensors. This means the navigator may not send different weights to individual sensors during a single instance t.

Honest-but-Curious Sensors We adopt the honest-but-curious attacker model for all involved sensors, meaning that they follow the localisation procedure correctly but may store or use any gained sensitive information.

We justify the global broadcast assumption by noting that any subset of sensors within the range of the navigator can be considered a group and treated as the global set during estimation, generalising the method, while the widespread use of cheap non-directional antennas supports the assumption of consistent broadcasts. The final assumption refers to the known problem of misbehaving sensors [lazosSeRLocSecureRangeindependent2004,bengalOutlierDetection2005], often requiring additional complicated detection mechanisms, and will not be considered in this work.

We are now ready to define the type of encryption scheme we want for the specified communication protocol and the security guarantees it should provide. We let a linear combination aggregation scheme be defined as a tuple of the four algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{CombEnc}, \mathsf{AggDec})$. These will be used by a trusted setup party, the navigator, and sensors $i \in \{1, \dots, n\}$. They are defined as follows.

$\mathsf{Setup}(\kappa)$ On input of security parameter κ, generate public parameters pub, the number of weights m, the navigator's public and private keys pk_0 and sk_0 and the sensor private keys sk_i, $i \in \{1, \dots, n\}$.

$\mathsf{Enc}(pk_0, x)$ The navigator and sensors can encrypt any value x with the navigator's public key pk_0 and obtain the encryption $\mathcal{E}_{pk_0}(x)$.

$\mathsf{CombEnc}(t, pk_0, sk_i, \mathcal{E}(\omega_1^{(t)}), \dots, \mathcal{E}(\omega_m^{(t)}), a_{i,1}^{(t)}, \dots, a_{i,m}^{(t)})$ At instance t, sensor i computes and obtains the encrypted linear combination denoted $l_i^{(t)} = \mathcal{E}_{pk_0, sk_i}(\sum_{j=1}^{m} a_{i,j}^{(t)} \omega_j^{(t)})$ using its secret key sk_i.

$\mathsf{AggDec}(t, pk_0, sk_0, l_1^{(t)}, \dots, l_n^{(t)})$ At instance t, the navigator computes the aggregation of linear combinations $\sum_{i=1}^{n} l_i^{(t)} = \sum_{i=1}^{n} \sum_{j=1}^{m} a_{i,j}^{(t)} \omega_j^{(t)}$ using its public and private keys pk_0, sk_0.

The security notions we want these algorithms to meet reflect the previously stated estimation privacy goals. The navigator should learn no information from individual sensors while sensors should learn no information from the navigator or any other sensors. In the context of the introduced communication protocol, this can be summarised as the following notions.

Indistinguishable Weights No colluding subset of sensors gains any new knowledge about the navigator weights $\omega_j^{(t)}$, $j \in \{1, \dots, m\}$ when receiving only their encryptions from the current and previous instances and having the ability to encrypt plaintexts of their choice.

Linear Combination Aggregator Obliviousness No colluding subset *excluding* the navigator gains additional information about the remaining sensor values to be weighted, $a_{i,j}^{(t)}$, $j \in \{1, \dots, m\}$, where sensor i is not colluding, given only encryptions of their linear combinations l_i from the current and previous instances. Any colluding subset *including* the navigator learns only the sum of all linear combinations weighted by weights of their choice, $\sum_{i=1}^n l_i^{(t)} = \sum_{i=1}^n \sum_{j=1}^m a_{i,j}^{(t)} \omega_j^{(t)}$.

While indistinguishable weights can be achieved by encrypting weights with an encryption scheme meeting the notion of Indistinguishability under the Chosen Plaintext Attack (IND-CPA) [katzIntroductionModernCryptography2008], the novel notion of Linear Combination Aggregator Obliviousness (LCAO) has been formalised as a typical cryptographic game between attacker and challenger in the appendix [app:lcao]. Lastly, we conclude the cryptographic problem definition with the following important remark.

Remark. A leakage function including weights from the navigator requires extra care to be taken when giving its definition. If an attacker compromises the navigator, they have control over the weights, and therefore the leakage function. We note that in the leakage function above, $\sum_{i=1}^n \sum_{j=1}^m a_{i,j}^{(t)} \omega_j^{(t)}$, an individual sum weighted by the same weight may be learnt by an attacker, *e.g.*, $\sum_{i=1}^n a_{i,1}^{(t)}$ given weights $(1, 0, \dots, 0)$, but that individual sensor values $a_{i,j}^{(t)}$ remain private due to the assumption of a consistent broadcast.

4.1.2. Estimation problem

The estimation problem we consider, for which we will reformulate communication to the protocol above, is localisation with range-only sensors. In this work, we will focus on the two-dimensional case for simplicity but will derive methods suitable for extension to a three-dimensional equivalent. The state that we wish to estimate must capture the navigator position, x and y, and may contain any other components relevant to the system. It is of the form

$$\underline{x} = \begin{bmatrix} x & y & \cdots \end{bmatrix}^\top . \tag{4.1}$$

This state evolves following some known system model, which at timestep k can be written as

$$\underline{x}_k = \underline{f}_k(\underline{x}_{k-1}, \underline{w}_k) , \tag{4.2}$$

with noise term $\underline{w}_k$. Measurements of $\underline{x}_k$ follow a measurement model dependent on sensor $i \in \{1, \dots, n\}$, given by

$$z_{k,i} = h_i(\underline{x}_k) + v_{k,i} , \tag{4.3}$$

with Gaussian measurement noises $v_{k,i} \sim \mathcal{N}(0, r_{k,i})$ and measurement function

$$h_i(\underline{x}) = \left\| \begin{bmatrix} x & y \end{bmatrix}^\top - \underline{s}_i \right\| \\ = \sqrt{(x - s_{x,i})^2 + (y - s_{y,i})^2}\,, \tag{4.4}$$

where

$$\underline{s}_i = \begin{bmatrix} s_{x,i} & s_{y,i} \end{bmatrix}^\top \tag{4.5}$$

is the location of sensor i.

We aim to provide a filter that estimates the navigator's state $\underline{x}_k$, at every timestep k, without learning sensor positions $\underline{s}_i$, measurements $z_{k,i}$ and measurement variances $r_{k,i}$ beyond the information in the corresponding aggregation leakage function. Similarly, sensors should not learn any information about current state estimates or any other sensor information. Leakage will be further discussed in section 4.2.4, but we note that from any sequential state estimates, following known models, some sensor information leakage can be computed by the navigator. In the context of our leakage function, we will show that this corresponds to the global sums of private sensor information, while individual, or subsets of sensors', information remains private. Similarly, corrupted sensors with access to one or more measurements can produce state estimates of their own, leaking information about navigator state estimates, however, the most accurate estimates, requiring all measurements, will always remain private to the navigator.

4.2. Confidential Range-Only Localisation

4.2.1. Private Linear Combination Aggregation Scheme

In this section, we introduce an encryption scheme meeting the desired security properties in section 4.1.1. The scheme is a combination of the Paillier and Joye-Libert schemes and provides encrypted weights meeting IND-CPA and encrypted aggregation meeting the notion of LCAO defined in section 4.1.1. Similarly to its constituents, the scheme bases its security on the DCRA and, as with the Joye-Libert scheme, requires a trusted party for initial key generation and distribution.

As aggregation is typically performed on scalar inputs, we extend our notation to the context of multidimensional estimation data by letting an instance $t_{k,\tau}$ uniquely capture the scalar aggregation during an estimation timestep k for a single element with position index τ. To achieve this in practice, any injective function can be used, such as the concatenation $t_{k,\tau} = k \parallel \tau$. The four algorithms defining our scheme are given as follows.

$\mathsf{Setup}(\kappa)$ On input parameter κ, generate two equal-length, sufficiently large, primes p and q, and compute $N = pq$. Define a hash function $H : \mathbb{Z} \to \mathbb{Z}^*_{N^2}$, choose the number of weights to combine, $m > 1$, and set public parameter $\mathsf{pub} = H$, navigator public key $pk_0 = N$ and navigator private key $sk_0 = (p, q)$. Sensor secret keys are generated by choosing sk_i, $i \in \{1, \dots, n-1\}$ uniformly from $\mathbb{Z}_{N^2}$ and setting the last key to $sk_n = -\sum_{i=1}^{n-1} sk_i$.

$\mathsf{Enc}(pk_0, x)$ Public-key encryption is computed by the Paillier encryption scheme with implicit generator $g = N + 1$. This is given by

$$\mathcal{E}_{pk_0}(x) = (N+1)^x \rho^N \pmod{N^2}, \tag{4.6}$$

for a randomly chosen $\rho \in \mathbb{Z}_N$.

$\mathsf{CombEnc}(t_{k,\tau}, pk_0, sk_i, \mathcal{E}_{pk_0}(\omega_1^{(k,\tau)}) \dots, a_{i,1}^{(k,\tau)} \dots)$ At the instance $t_{k,\tau}$, encrypted linear combination is given by

$$l_i^{(k,\tau)} = H(t_{k,\tau})^{sk_i} \prod_{j=1}^{m} \mathcal{E}_{pk_0}(\omega_j^{(k,\tau)})^{a_{i,j}^{(k,\tau)}} \pmod{N^2}, \tag{4.7}$$

and makes use of the homomorphic property (**??**). Correctness follows from

$$\begin{aligned} l_i^{(k,\tau)} &= H(t_{k,\tau})^{sk_i} \prod_{j=1}^{m} \mathcal{E}_{pk_0}(\omega_j^{(k,\tau)})^{a_{i,j}^{(k,\tau)}} \pmod{N^2} \\ &= H(t_{k,\tau})^{sk_i} \prod_{j=1}^{m} \mathcal{E}_{pk_0}(a_{i,j}^{(k,\tau)} \omega_j^{(k,\tau)}) \pmod{N^2} \\ &= H(t_{k,\tau})^{sk_i} \cdot \\ &\quad \prod_{j=1}^{m} (N+1)^{a_{i,j}^{(k,\tau)} \omega_j^{(k,\tau)}} \rho_j^N \pmod{N^2} \\ &= H(t_{k,\tau})^{sk_i} \cdot \\ &\quad (N+1)^{\sum_{j=1}^{m} a_{i,j}^{(k,\tau)} \omega_j^{(k,\tau)}} \tilde{\rho}_i^N \pmod{N^2}, \end{aligned}$$

for some values $\rho_j \in \mathbb{Z}_N$, $j \in \{1, \dots, m\}$ and $\tilde{\rho}_i = \prod_{j=1}^{m} \rho_j$. Here, $\tilde{\rho}_i^N$ and $H(t_{k,\tau})^{sk_i}$ can be considered the noise terms corresponding to the two levels of encryption from pk_0 and sk_i, respectively.

$\mathsf{AggDec}(t_{k,\tau}, pk_0, sk_0, l_1^{(k,\tau)}, \dots, l_n^{(k,\tau)})$ Aggregation is computed as $l^{(k,\tau)} = \prod_{i=1}^{n} l_i^{(k,\tau)} \pmod{N^2}$, removing the aggregation noise terms, and is followed by Paillier scheme decryption

$$\begin{aligned} &\sum_{i=1}^{n} \sum_{j=1}^{m} a_{i,j}^{(k,\tau)} \omega_j^{(k,\tau)} = \\ &\qquad \frac{L((l^{(k,\tau)})^{\lambda} \pmod{N^2})}{L((N+1)^{\lambda} \pmod{N^2})} \pmod{N}, \end{aligned} \tag{4.8}$$

with $\lambda = \mathsf{lcm}(p-1, q-1)$ and $L(\psi) = \frac{\psi - 1}{N}$. The correctness of the aggregation can be seen from

$$
\begin{aligned}
l^{(k,\tau)} &= \prod_{i=1}^{n} H(t_{k,\tau})^{sk_i} \cdot \\
&\qquad (N+1)^{\sum_{j=1}^{m} a_{i,j}^{(k,\tau)} \omega_j^{(k,\tau)}} \tilde{\rho}_i^N \pmod{N^2} \\
&= H(t_{k,\tau})^{\sum_{i=1}^{n} sk_i} \cdot \\
&\qquad \prod_{i=1}^{n} (N+1)^{\sum_{j=1}^{m} a_{i,j}^{(k,\tau)} \omega_j^{(k,\tau)}} \tilde{\rho}_i^N \pmod{N^2} \\
&= (N+1)^{\sum_{i=1}^{n} \sum_{j=1}^{m} a_{i,j}^{(k,\tau)} \omega_j^{(k,\tau)}} \tilde{\rho}^N \pmod{N^2},
\end{aligned}
$$

for some values $\tilde{\rho}_i \in \mathbb{Z}_N$, $i \in \{1, \dots, n\}$ and $\tilde{\rho} = \prod_{i=1}^{n} \tilde{\rho}_i$.

Additionally, we note that in the above construction, all weights $\omega_j^{(k,\tau)}$ and values $a_{i,j}^{(k,\tau)}$ are integers and the resulting linear combinations and aggregation are computed modulo N.

The security proof of this scheme must both show that encrypted weights meet IND-CPA and that encrypted aggregation meets LCAO. As weights are encrypted with the Paillier encryption scheme, the first requirement is already met. To show that aggregation meets LCAO, a reduction proof is given in the appendix [app:proof].

Remark. Given the construction of the scheme above, it can be seen that any weights $\omega_j^{(k,\tau)}$, whose values are known at each sensor, do not need to be broadcast by the navigator. In this case, sensors can replace

$$
\mathcal{E}_{pk_0}(\omega_j^{(k,\tau)})^{a_{i,j}^{(k,\tau)}} = (N+1)^{\omega_j^{(k,\tau)} a_{i,j}^{(k,\tau)}} \rho_j^N \pmod{N^2} \tag{4.9}
$$

in (4.7), by

$$
(N+1)^{\omega_j^{(k,\tau)} a_{i,j}^{(k,\tau)}} \pmod{N^2}. \tag{4.10}
$$

This is due to the removal of ρ_j^N terms during decryption and can be used to reduce the navigator's broadcast communication cost by the number of weights $\omega_j^{(k,\tau)}$ that do not hold any information private to the navigator and are known by the sensors in advance.

4.2.2. Privacy-Preserving Localisation

With a concrete scheme meeting the LCAO notion, we can now put forward a localisation filter with communication that can be reformulated to the required protocol. To produce an estimate of the state $\underline{x}_k$, we make use of an algebraic reformulation of the Extended Kalman Filter (EKF), the Extended Information Filter (EIF) [maybeckStochasticModelsEstimation1982], which reduces the filter update step to a single summation. The EIF update step requires the predicted state estimate $\hat{\underline{x}}_{k|k-1}$ and estimate covariance

$\mathbf{P}_{k|k-1}$ in the information vector and matrix forms

$$\hat{\underline{y}}_{k|k-1} = \mathbf{P}_{k|k-1}^{-1}\hat{\underline{x}}_{k|k-1} \quad \text{and} \quad \mathbf{Y}_{k|k-1} = \mathbf{P}_{k|k-1}^{-1}\,, \tag{4.11}$$

respectively. In this form, the update equations for n sensor measurements at time k, with measurement models (4.3), are given by

$$\begin{aligned} \hat{\underline{y}}_{k|k} &= \hat{\underline{y}}_{k|k-1} + \\ &\sum_{i=1}^{n} \mathbf{H}_{k,i}^{\top} r_i^{-1} \left(z_{k,i} - h_i(\hat{\underline{x}}_{k|k-1}) + \mathbf{H}_{k,i}\hat{\underline{x}}_{k|k-1} \right) \end{aligned} \tag{4.12}$$

and

$$\mathbf{Y}_{k|k} = \mathbf{Y}_{k|k-1} + \sum_{i=1}^{n} \mathbf{H}_{k,i}^{\top} r_i^{-1} \mathbf{H}_{k,i}\,, \tag{4.13}$$

with Jacobians

$$\mathbf{H}_{k,i} = \left.\frac{\partial h_i}{\partial \underline{x}}\right|_{\hat{\underline{x}}_{k|k-1}} \tag{4.14}$$

for sensors $i \in \{1, \dots, n\}$. After converting the updated information vector and matrix back to state estimate $\hat{\underline{x}}_{k|k}$ and estimate covariance $\mathbf{P}_{k|k}$, the filter's prediction step can be computed by the navigator locally using any suitable filter for the known system model (4.2).

In the form above, at every timestep k, all sensitive sensor information required for state estimation is captured in the measurement vector

$$\underline{i}_{k,i} = \mathbf{H}_{k,i}^{\top} r_i^{-1} \left(z_{k,i} - h_i(\hat{\underline{x}}_{k|k-1}) + \mathbf{H}_{k,i}\hat{\underline{x}}_{k|k-1} \right) \tag{4.15}$$

and the measurement matrix

$$\mathbf{I}_{k,i} = \mathbf{H}_{k,i}^{\top} r_i^{-1} \mathbf{H}_{k,i}\,, \tag{4.16}$$

namely, their measurements $z_{k,i}$, measurement variances $r_{k,i}$ and locations $\underline{s}_i$; captured in measurement functions h_i and Jacobians $\mathbf{H}_{k,i}$. To compute $\underline{i}_{k,i}$ and $\mathbf{I}_{k,i}$, however, the current predicted state estimate $\hat{\underline{x}}_{k|k-1}$ is also required (in h_i and $\mathbf{H}_{k,i}$). Therefore, our goal is to rearrange (4.15) and (4.16) as a linear combination of functions of $\hat{\underline{x}}_{k|k-1}$ (the navigator weights) computable at each sensor i, to be subsequently aggregated at the navigator. Application of the linear combination aggregation scheme proposed in turn guarantees that sensors do not learn the navigator state, and the navigator learns only the aggregation required for updating its state estimate in (4.12) and (4.13).

Range Measurement Modification

The first thing we notice when wanting to rearrange $\underline{i}_{k,i}$ and $\mathbf{I}_{k,i}$ to a linear combination of functions of $\hat{\underline{x}}_{k|k-1}$, is that h_i cannot be rearranged in this way due to the present square-root. Similarly, the Jacobian of h_i at $\underline{\hat{x}}_{k|k-1}$,

$$\mathbf{H}_{k,i} = \begin{bmatrix} \frac{\hat{x}_{k|k-1}-s_{x,i}}{\sqrt{(\hat{x}_{k|k-1}-s_{x,i})^2+(\hat{y}_{k|k-1}-s_{y,i})^2}} \\ \frac{\hat{y}_{k|k-1}-s_{y,i}}{\sqrt{(\hat{x}_{k|k-1}-s_{x,i})^2+(\hat{y}_{k|k-1}-s_{y,i})^2}} \\ 0 \\ \vdots \end{bmatrix}^\top , \tag{4.17}$$

cannot be either. We, therefore, consider the modified measurement functions

$$h'_i(\underline{x}) = h_i(\underline{x})^2 . \tag{4.18}$$

A measurement function in this form allows rearrangement of h'_i and the corresponding Jacobian $\mathbf{H}'_{k,i}$ to a linear combination of powers of location elements in $\underline{\hat{x}}_{k|k-1}$, as

$$\begin{aligned} h'_i(\underline{x}) &= \left\| \begin{bmatrix} x & y \end{bmatrix}^\top - \underline{s}_i \right\|^2 \\ &= (x - s_{x,i})^2 + (y - s_{y,i})^2 \\ &= x^2 + y^2 - 2s_{x,i}x - 2s_{y,i}y + s_{x,i}^2 + s_{y,i}^2 \end{aligned} , \tag{4.19}$$

and

$$\mathbf{H}'_{k,i} = \begin{bmatrix} 2\hat{x}_{k|k-1} - 2s_{x,i} \\ 2\hat{y}_{k|k-1} - 2s_{y,i} \\ 0 \\ \vdots \end{bmatrix}^\top . \tag{4.20}$$

Here, h'_i and $\mathbf{H}'_{k,i}$ are linear combinations of $\hat{x}^2_{k|k-1}$, $\hat{y}^2_{k|k-1}$, $\hat{x}_{k|k-1}$ and $\hat{y}_{k|k-1}$. To show how the corresponding modified measurement vectors $\underline{i}'_{k,i}$ and matrices $\mathbf{I}'_{k,i}$ can be similarly rearranged and used for localisation, we also require the existence of measurements following a modified measurement model of the form

$$z'_{k,i} = h'_i(\underline{x}_k) + v'_{k,i} , \tag{4.21}$$

where $z'_{k,i}$ is the modified measurement, and noise term $v'_{k,i}$ is zero-mean and has a known variance $r'_{k,i}$.

Computing $z'_{k,i}$ and its variance $r'_{k,i}$ from the original measurements $z_{k,i}$ are complicated by the noise term $v_{k,i} \sim \mathcal{N}(0, r_{k,i})$, and simply squaring the original range measurements produces

$$\begin{aligned} z^2_{k,i} &= (h_i(\underline{x}_k) + v_{k,i})^2 \\ &= h'_i(\underline{x}_k) + 2h_i(\underline{x}_k)v_{k,i} + v^2_{k,i} , \end{aligned} \tag{4.22}$$

with a new noise term $2h_i(\underline{x}_k)v_{k,i} + v^2_{k,i}$, now dependent on the measurement function h_i, and no longer zero-mean. We can compute the mean of this new noise term (a function of the Gaussian term $v_{k,i}$) as $\mathsf{E}[2h_i(\underline{x}_k)v_{k,i} + v_{k,i}^2] = r_{k,i}$ and mean-adjust modified measurements as

$$\begin{aligned} z'_{k,i} &= z_{k,i}^2 - r_{k,i} \\ &= h_i(\underline{x}_k)^2 + 2h_i(\underline{x}_k)v_{k,i} + v_{k,i}^2 - r_{k,i} \\ &= h'_i(\underline{x}_k) + v'_{k,i} \,, \end{aligned} \tag{4.23}$$

with now zero-mean noise $v'_{k,i} = 2h_i(\underline{x}_k)v_{k,i} + v_{k,i}^2 - r_{k,i}$. The noise in this case (again a function of $v_{k,i}$) has variance

$$\mathsf{Var}[v'_{k,i}] = 4h_i(\underline{x}_k)^2 r_{k,i} + 2r_{k,i}^2 \tag{4.24}$$

and is also dependent on h_i. To use the modified measurement (4.23) with the EIF, we require an estimate for $\mathsf{Var}[v'_{k,i}]$ at the sensor as well. Additionally, a conservative estimate (*i.e.*, a larger variance resulting in less confidence in measurements) is desirable to reduce filter divergence. While the naive approach, replacing $h_i(\underline{x}_k)$ with $z_{k,i}$ in (4.24), may not provide a conservative estimate when $z_{k,i}^2 < h_i(\underline{x}_k)^2$, the Gaussianity of $v_{k,i}$ can be exploited to provide a conservative estimate with 95% confidence by shifting the replacement term $z_{k,i}$ by two of its standard deviations $\sqrt{r_{k,i}}$. The modified measurement's variance at timestep k can therefore be conservatively approximated by

$$\begin{aligned} r'_{k,i} &= 4(z_{k,i} + 2\sqrt{r_{k,i}})^2 r_{k,i} + 2r_{k,i}^2 \\ &\gtrapprox \mathsf{Var}[v'_{k,i}] \,, \end{aligned} \tag{4.25}$$

at each sensor i.

The modified measurement model (4.21) can now be used for localisation, when measurements are modified by (4.23) and their new variance estimated with (4.25).

Localisation

To complete the EIF update as a linear combination aggregation, modified vectors $\underline{i}'_{k,i}$ and matrices $\mathbf{I}'_{k,i}$, using the modified measurement model (4.21), can be rearragned as follows.

$$\begin{aligned} \underline{i}'_{k,i} &= \mathbf{H}'^{\top}_{k,i} r'^{-1}_{k,i} (z'_{k,i} - h'_i(\hat{\underline{x}}_{k|k-1}) + \mathbf{H}'_{k,i}\hat{\underline{x}}_{k|k-1}) \\ &= \begin{bmatrix} \alpha_i^{(k,1)} & \alpha_i^{(k,2)} & 0 & \cdots \end{bmatrix}^\top \,, \end{aligned} \tag{4.26}$$

with

$$\begin{aligned}
\alpha_i^{(k,1)} &= (2r_{k,i}'^{-1})\hat{x}_{k|k-1}^3 + (2r_{k,i}'^{-1})\hat{x}_{k|k-1}\hat{y}_{k|k-1}^2 \\
&\quad + (-2r_{k,i}'^{-1}s_{x,i})\hat{x}_{k|k-1}^2 + (-2r_{k,i}'^{-1}s_{x,i})\hat{y}_{k|k-1}^2 \\
&\quad + (2r_{k,i}'^{-1}z_{k,i}')\hat{x}_{k|k-1} + (-2r_{k,i}'^{-1}s_{x,i}^2)\hat{x}_{k|k-1} \\
&\quad + (-2r_{k,i}'^{-1}s_{y,i}^2)\hat{x}_{k|k-1} + (2r_{k,i}'^{-1}s_{x,i}^3) \\
&\quad + (2r_{k,i}'^{-1}s_{x,i}s_{y,i}^2) + (-2r_{k,i}'^{-1}s_{x,i}z_{k,i}') \text{ and} \\
\alpha_i^{(k,2)} &= (2r_{k,i}'^{-1})\hat{y}_{k|k-1}^3 + (2r_{k,i}'^{-1})\hat{x}_{k|k-1}^2\hat{y}_{k|k-1} \\
&\quad + (-2r_{k,i}'^{-1}s_{y,i})\hat{x}_{k|k-1}^2 + (-2r_{k,i}'^{-1}s_{y,i})\hat{y}_{k|k-1}^2 \\
&\quad + (2r_{k,i}'^{-1}z_{k,i}')\hat{y}_{k|k-1} + (-2r_{k,i}'^{-1}s_{x,i}^2)\hat{y}_{k|k-1} \\
&\quad + (-2r_{k,i}'^{-1}s_{y,i}^2)\hat{y}_{k|k-1} + (2r_{k,i}'^{-1}s_{y,i}s_{x,i}^2) \\
&\quad + (2r_{k,i}'^{-1}s_{y,i}^3) + (-2r_{k,i}'^{-1}s_{y,i}z_{k,i}')\,,
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{I}_{k,i}' &= \mathbf{H}_{k,i}'^{\top}r_{k,i}'^{-1}\mathbf{H}_{k,i}' \\
&= \begin{bmatrix} \alpha_i^{(k,3)} & \alpha_i^{(k,4)} & 0 & \cdots \\ \alpha_i^{(k,5)} & \alpha_i^{(k,6)} & 0 & \cdots \\ 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}\,,
\end{aligned} \tag{4.27}$$

with

$$\begin{aligned}
\alpha_i^{(k,3)} &= (4r_{k,i}'^{-1})\hat{x}_{k|k-1}^2 + (-8r_{k,i}'^{-1}s_{x,i})\hat{x}_{k|k-1} \\
&\quad + (4r_{k,i}'^{-1}s_{x,i}^2)\,, \\
\alpha_i^{(k,4)} &= (4r_{k,i}'^{-1})\hat{x}_{k|k-1}\hat{y}_{k|k-1} + (-4r_{k,i}'^{-1}s_{y,i})\hat{x}_{k|k-1} \\
&\quad + (-4r_{k,i}'^{-1}s_{x,i})\hat{y}_{k|k-1} + (4r_{k,i}'^{-1}s_{x,i}s_{y,i})\,, \\
\alpha_i^{(k,5)} &= \alpha_i^{(k,4)} \text{ and} \\
\alpha_i^{(k,6)} &= (4r_{k,i}'^{-1})\hat{y}_{k|k-1}^2 + (-8r_{k,i}'^{-1}s_{y,i})\hat{y}_{k|k-1} \\
&\quad + (4r_{k,i}'^{-1}s_{y,i}^2)\,.
\end{aligned}$$

The above rearrangements give $\underline{i}_{k,i}'$ and $\mathbf{I}_{k,i}'$ as linear combinations of elements in

$$\begin{aligned}
&\{\hat{x}_{k|k-1}^3,\ \hat{y}_{k|k-1}^3,\ \hat{x}_{k|k-1}^2\hat{y}_{k|k-1},\ \hat{x}_{k|k-1}\hat{y}_{k|k-1}^2, \\
&\quad \hat{x}_{k|k-1}^2,\ \hat{y}_{k|k-1}^2,\ \hat{x}_{k|k-1}\hat{y}_{k|k-1},\ \hat{x}_{k|k-1},\ \hat{y}_{k|k-1}\}\,,
\end{aligned} \tag{4.28}$$

which captures all of the private state information in $\hat{\underline{x}}_{k|k-1}$ required at the sensors. The corresponding EIF update steps (4.12) and (4.13) then become

$$\hat{\underline{y}}_{k|k} = \hat{\underline{y}}_{k|k-1} + \sum_{i=1}^{n} \underline{i}'_{k,i} \tag{4.29}$$

and

$$\mathbf{Y}_{k|k} = \mathbf{Y}_{k|k-1} + \sum_{i=1}^{n} \mathbf{I}'_{k,i} \,, \tag{4.30}$$

respectively.

Remark. The above has been derived for two-dimensional localisation but can be similarly derived for the three-dimensional case. However, the number of weights increases combinatorially with the number of dimensions, thus affecting the cost of communication as well.

Pseudocode

Measurement modification, real number encoding and linear combination aggregation are all required to compute the modified EIF from the previous section in a privacy-preserving manner. In this section, we summarise this entire localisation process and give the pseudocode for its execution. For brevity, we will assume ϕ and $M = N$ from section [subsec:encoding] to be public information and thus simplify the encoding notation $\mathsf{E}_{N,\phi,d}(\cdot)$ to $\mathsf{E}_d(\cdot)$. The privacy-preserving localisation filter consists of the following steps.

Setup The Setup algorithm from section 4.2.1 is run only once by a trusted party, N and H are made public, and the navigator and sensor secret keys, $sk_0 = \lambda = \mathsf{lcm}(p-1, q-1)$ and sk_i, $i \in \{1, \dots, n\}$, are distributed accordingly.

Prediction At each timestep k, the navigator computes the prediction of the current state and its covariance with a local filter before encrypting weights (4.28) with algorithm Enc and broadcasting them to the sensors. This is given by algorithm 4.

Measurement At each timestep k, sensors modify their measurements with (4.23) and (4.25) before computing encryptions of $\underline{i}'_{k,i}$ and $\mathbf{I}'_{k,i}$ using algorithm CombEnc for each element and sending them back to the navigator. This is given by algorithm 5.

Update At each timestep k, the navigator aggregates and decrypts recieved measurement vectors and matrices with algorithm AggDec, before computing the EIF update equations (4.29) and (4.30). This is given by algorithm 6.

Algorithms 4, 5 and 6 have also been summarised graphically in figure 4.2. Here, for brevity, $\mathcal{E}_{pk_0,sk_i}(\cdot)$ and $\mathsf{E}_d(\cdot)$ denote elementwise operations with the same parameters.

Algorithm 4 Navigator Prediction

procedure PREDICTION($\hat{\underline{x}}_{k-1|k-1}$, $\mathbf{P}_{k-1|k-1}$, N)
  Compute $\hat{\underline{x}}_{k|k-1}$ with a local filter
  Compute $\mathbf{P}_{k|k-1}$ with a local filter
  Compute $\mathsf{E}_0(\hat{x}^3_{k|k-1})$ by (**??**)
  Compute $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}^3_{k|k-1}))$ by (4.6)
  Broadcast $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}^3_{k|k-1}))$ to sensors
  for Remaining weights in (4.28) **do**
    Broadcast weight in the form above
  end for
  return $\hat{\underline{x}}_{k|k-1}, \mathbf{P}_{k|k-1}$
end procedure

Algorithm 5 Measurement at Sensor i

procedure MEASUREMENT(i, $s_{x,i}$, $s_{y,i}$, $r_{k,i}$, N, H)
  Measure $z_{k,i}$
  Compute $z'_{k,i}$ by (4.23)
  Compute $r'_{k,i}$ by (4.25)
  Recieve $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}^3_{k|k-1}))$
  for Remaining weights in (4.28) **do**
    Recieve weight in the form above
  end for
  Let $\boldsymbol{\alpha}_i^{(k,\tau)}$ represent the encryption of $\alpha_i^{(k,\tau)}$ in (4.26) and (4.27)
  $\boldsymbol{\alpha}_i^{(k,1)} \leftarrow \mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}^3_{k|k-1}))^{\mathsf{E}_0(2r'^{-1}_{k,i})}\cdot$
  $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}_{k|k-1}\hat{y}^2_{k|k-1}))^{\mathsf{E}_0(2r'^{-1}_{k,i})}\cdot$
  $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}^2_{k|k-1}))^{\mathsf{E}_0(-2r'^{-1}_{k,i}s_{x,i})}\cdot$
  $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{y}^2_{k|k-1}))^{\mathsf{E}_0(-2r'^{-1}_{k,i}s_{x,i})}\cdot$
  $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}_{k|k-1}))^{\mathsf{E}_0(2r'^{-1}_{k,i}z'_{k,i})}\cdot$
  $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}_{k|k-1}))^{\mathsf{E}_0(-2r'^{-1}_{k,i}s^2_{x,i})}\cdot$
  $\mathcal{E}_{pk_0}(\mathsf{E}_0(\hat{x}_{k|k-1}))^{\mathsf{E}_0(-2r'^{-1}_{k,i}s^2_{y,i})}\cdot$
  $(N+1)^{\mathsf{E}_1(2r'^{-1}_{k,i}s^3_{x,i})}(N+1)^{\mathsf{E}_1(2r'^{-1}_{k,i}s_{x,i}s^2_{y,i})}\cdot$
  $(N+1)^{\mathsf{E}_1(-2r'^{-1}_{k,i}s_{x,i}z'_{k,i})}H(k \,\|\, 1) \pmod{N^2}$
  Compute remaining $\boldsymbol{\alpha}_i^{(k,\tau)}$ using (4.26), (4.27), (4.7) and the remark from section 4.2.1 in the form above
  for $\tau \leftarrow 1$ to 6 **do**
    Send $\boldsymbol{\alpha}_i^{(k,\tau)}$ to the navigator
  end for
end procedure

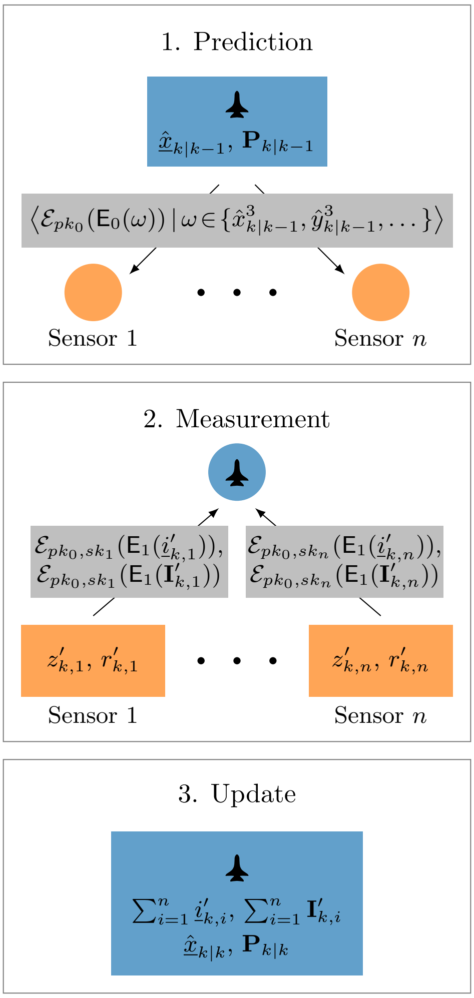


Figure 4.2.: Procedure at timestep k for the proposed privacy-preserving EIF.

Algorithm 6 Navigator Update

procedure UPDATE($\underline{\hat{x}}_{k|k-1}$, $\mathbf{P}_{k|k-1}$, N, λ)
    for $\tau \leftarrow 1$ to 6 **do**
        Receive $\boldsymbol{\alpha}_i^{(k,\tau)}$ from each sensor $i \in \{1, \dots, n\}$
    end for
    Let $\boldsymbol{\alpha}^{(k,\tau)}$ represent an encryption of $\sum_{i=1}^{n} \alpha_i^{(k,\tau)}$
    for $\tau \leftarrow 1$ to 6 **do**
        $\boldsymbol{\alpha}^{(k,\tau)} \leftarrow \prod_{i=1}^{n} \boldsymbol{\alpha}_i^{(k,\tau)}$
        Compute $\mathcal{D}_{sk_0}(\boldsymbol{\alpha}^{(k,\tau)})$ with λ by (4.8)
        Compute $\mathsf{E}_1^{-1}(\mathcal{D}_{sk_0}(\boldsymbol{\alpha}^{(k,\tau)}))$ by (**??**)
    end for
    Construct $\sum_{i=1}^{n} \underline{i}'_{k,i}$ and $\sum_{i=1}^{n} \mathbf{I}'_{k,i}$ from decoded decryptions above
    $\underline{\hat{y}}_{k|k} \leftarrow \mathbf{P}_{k|k-1}^{-1} \underline{\hat{x}}_{k|k-1} + \sum_{i=1}^{n} \underline{i}'_{k,i}$
    $\mathbf{Y}_{k|k} \leftarrow \mathbf{P}_{k|k-1}^{-1} + \sum_{i=1}^{n} \mathbf{I}'_{k,i}$
    $\underline{\hat{x}}_{k|k} \leftarrow \mathbf{Y}_{k|k}^{-1} \underline{\hat{y}}_{k|k}$
    $\mathbf{P}_{k|k} \leftarrow \mathbf{Y}_{k|k}^{-1}$
    return $\underline{\hat{x}}_{k|k}, \mathbf{P}_{k|k}$
end procedure

4.2.3. Solvable Sub-Class of Non-Linear Measurement Models

4.2.4. Security Analysis

With the privacy-preserving EIF defined in the previous section, we can now interpret the aggregation leakage of an LCAO scheme in the context of range sensor localisation. The leakage function from the AggDec algorithm corresponds to the information vector and matrix sums, $\sum_{i=1}^{n} \underline{i}'_{k,i}$ and $\sum_{i=1}^{n} \mathbf{I}'_{k,i}$, respectively. However, recalling that a compromised navigator can learn the individual sums weighted by the same weight, $\{\sum_{i=1}^{n} 2r_{k,i}^{-1},\ \sum_{i=1}^{n} -r_{k,i}^{-1} s_{x,i},\ \sum_{i=1}^{n} -2r_{k,i}^{-1} s_{x,i},\ \dots\}$ can be leaked as well. From this leakage, we can see that private sensor information, $z'_{k,i}$, $r'_{k,i}$ and $\underline{s}_i$, is present only in their complete sums

$$\sum_{i=1}^{n} z'_{k,i}\,,\ \sum_{i=1}^{n} r'_{k,i}\,,\ \sum_{i=1}^{n} s_{x,i} \text{ and } \sum_{i=1}^{n} s_{y,i}\,, \tag{4.31}$$

which in practice correspond to their averages. Therefore, in the context of our proposed localisation method, LCAO leakage corresponds to the averages of sensor private information, while individual sensor information remains private.

4.2.5. Simulation

As well as having shown the theoretical backing for the security of our scheme, we have simulated the proposed localisation method to evaluate its performance. A two-dimensional, linear, constant velocity process model,

$$\underline{x}_k = \begin{bmatrix} 1 & 0 & 0.5 & 0 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \underline{x}_{k-1} + \underline{w}_k \,,$$

where noise term $\underline{w}_k \sim \mathcal{N}(\underline{0}, \mathbf{Q})$ and

$$\mathbf{Q} = \frac{1}{10^3} \cdot \begin{bmatrix} 0.4 & 0 & 1.3 & 0 \\ 0 & 0.4 & 0 & 1.3 \\ 1.3 & 0 & 5.0 & 0 \\ 0 & 1.3 & 0 & 5.0 \end{bmatrix} \,,$$

was simulated and tracked with the algorithms in section 4.2.2, using a linear Kalman filter for the navigator's local state prediction. Code was written in the C programming language using the MPI library [theopenmpiprojectOpenMPI2020] to support asynchronous computations by the sensors and the navigator. The MG1 mask generation function and the SHA256 hash function, from the OpenSSL library [theopensslprojectOpenSSL2020], were used to implement the required hash function H, and the Libpaillier library [bethencourtLibpaillier2010] was used for the Paillier encryption scheme. Additionally, GNU libraries, GSL [thegsldevelopmentteamGSLGNUScientific2019] and GMP [granlundGMPGNUMultiple2020], were used for algebraic operations and multiple-precision encoded integers, respectively. All execution was performed on a 3.33GHz Xeon W3680 CPU, running on the Windows Subsystem for Linux (WSL).

We have considered multiple sensor layouts, each with four sensors, to capture the dependence of estimated modified measurement variances $r'_{k,i}$ on the original measurements $z_{k,i}$. These layouts of varying sensor distances are shown next to the simulation initial state and a sample track in figure 4.3. To demonstrate the accuracy of the method, we have compared the root mean square error (RMSE) of the privacy-preserving filter to the standard EIF using unmodified measurements, which is algebraically equivalent to the EKF typically used in industry for linearising non-linear state estimation. Estimation in each layout from figure 4.3 consisted of 50 filter iterations and was run 1000 times. Unmodified measurement variances were taken as $r_{k,i} = 5$ for all $k > 0$ and a large fractional precision factor, $\phi = 2^{32}$, was chosen. The results can be seen in figure 4.4. From these results, we can see a strong similarity in filter performance between the privacy-preserving method and that of the traditional EIF. We can also see that the varying average distances between sensors and the navigator have little impact on the differences in performance. We attribute this similarity in RMSE to the conservativeness of estimated modified measurement variances $r'_{k,i}$, eliminating additional filter divergences, and to the high fractional precision factor, keeping computations consistent with the floating-point arithmetic of the EIF.

In addition to filter error, computational performance is important to consider when relying on cryptographic methods. Figure 4.5 shows the averages of 10 execution times

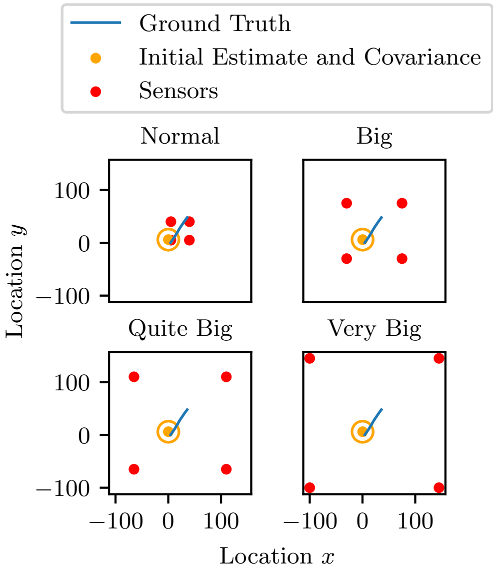


Figure 4.3.: Different simulation layouts with varying distances between navigator and sensors.

when varying the numbers of sensors and key sizes (bit lengths of N). Here, increasing the number of sensors primarily affects the number of inter-process communications and aggregation steps due to the asynchronous implementation. We can see that the predominant computational costs stem from cryptographic computations and are directly dependent on the chosen key size. In practice, choosing a key size should take into account the duration of secrecy and the secret key lifetime. When relying on the DCRA for security, the current recommendation for encrypting government documents is the use of 2048 bit length keys [barkerRecommendationPairwiseKey2019]. For our implementation and aforementioned hardware, this results in a filter update roughly every $1.7s$. In a scenario where sensors are mobile and past navigations can be made public, reduced key sizes can be considered, while a further decrease in computation time could be achieved with code optimisations and more powerful hardware.

4.3. Conclusions on Confidential Distributed Non-Linear Measurement Fusion

We have presented a localisation filter in the presence of range-only sensors, which preserves both navigator and sensor privacies. A suitable cryptographic scheme has been introduced and a filter implementation compared and evaluated. Privacy-preserving range-only localisation is suitable for use in environments where sensor networks are untrusted or location is considered private and we hope to extend the method to broader measurement models in the future. Additional future work includes exploring more computationally efficient encryption schemes, the security implications of sensors that are

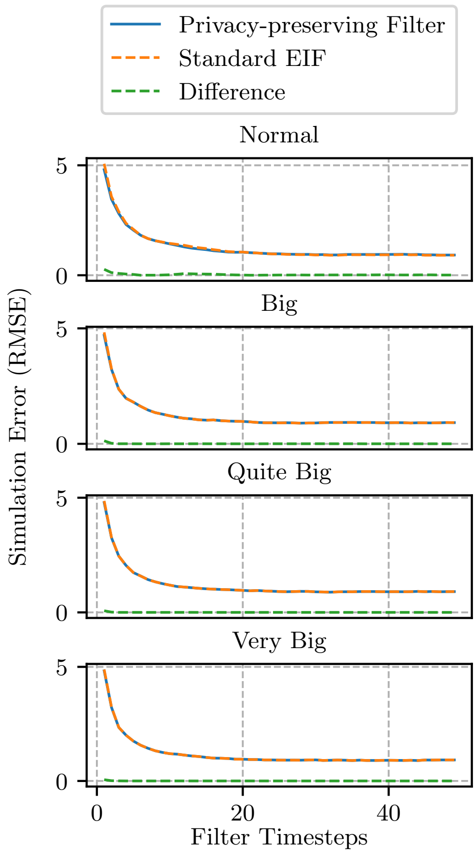


Figure 4.4.: Average RMSE of our privacy-preserving filter and the standard EIF for different layouts.

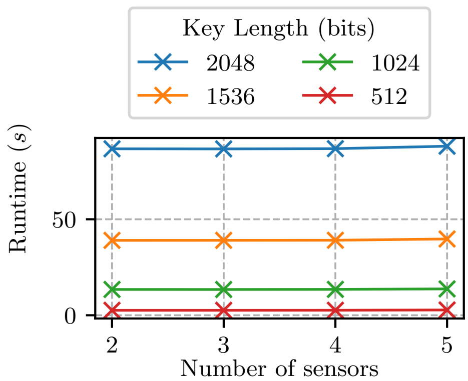


Figure 4.5.: Runtimes for varying key sizes and numbers of sensors.

not only honest-but-curious and expanding the LCAO notion to enforce the consistent broadcast assumption.

5. Provable Estimation Performances

5.1. Problem Formulation

In this work, we consider the estimation scenario where system and measurement models are known and stochastic, and state estimators are either privileged, when holding a secret key, or unprivileged, without. Our goal is to develop a scheme that quantifies and cryptographically guarantees the difference between their estimation errors when models are Gaussian and linear.

To capture the aim of creating a privileged and unprivileged estimator, we must first define how to assess estimation advantage between them, and which algorithms are required to characterize a privileged estimation scheme. In this section, we give the relevant formal definitions for security, followed by the system and measurement models considered in this work.

5.1.1. Formal Cryptographic Problem

While we later introduce assumptions on the system and measurement models, it is more practical to define a broader security notion that can be satisfied under arbitrary specified conditions on the models. This allows the use of the security notion in future literature and is more in line with typical cryptographic practice. Afterward, we will show that our proposed scheme meets this security notion under the specific Gaussian and linear model assumptions.

Typical formal cryptographic security notions are given in terms of probabilistic polynomial-time (PPT) attackers and capture desired privacy properties as well as attacker capabilities [Ch. 3][katzIntroductionModernCryptography2008]. The most commonly desired privacy property, cryptographic indistinguishability, is not suitable for our estimation scenario due to our desire for unprivileged estimators to gain *some* information from measurements. Instead, we will define security in terms of a time series of covariances, given arbitrary known models, such that the difference in estimation error between estimators with and without the secret key is bounded by the series at all times.

To formalize this, we introduce the following notations and definitions. We assume the existence of an arbitrary process (not necessarily Gaussian or linear) following a known system model exactly, with the state at time step k denoted by $\underline{x}_k \in \mathbb{R}^n$ and model parameters $\mathcal{M}_S$. Similarly, we assume the existence of a means of process measurement following a known measurement model exactly, with the measurement at time step k denoted by $\underline{y}_k \in \mathbb{R}^m$ and model parameters $\mathcal{M}_M$. We can now define a relevant scheme.

Definition 5.1.1. A *privileged estimation scheme* is a pair of probabilistic algorithms $(\mathsf{Setup}, \mathsf{Noise})$, given by

$\mathsf{Setup}(\mathcal{M}_S, \mathcal{M}_M, \kappa)$ On the input of models $\mathcal{M}_S$ and $\mathcal{M}_M$, and the security parameter κ, public parameters pub and a secret key sk are created.

$\mathsf{Noise}(\mathsf{pub}, \mathsf{sk}, k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1, \dots, \underline{y}_k)$ On input of public parameters pub, secret key sk, time step k, models $\mathcal{M}_S$ and $\mathcal{M}_M$, and measurements $\underline{y}_1, \dots, \underline{y}_k$, a noisy measurement $\underline{y}'_k$ (with no required model constraints) is created.

In addition to the scheme above, we also give the following definitions to help formalize our desired security notion.

Definition 5.1.2. An *estimator* is any probabilistic algorithm that produces a guess of the state $\underline{x}_k$ for a given time step k.

Definition 5.1.3. A *negligible covariance function*,

$$\mathsf{neglCov}_m(\kappa) : \mathbb{N} \to \mathbb{R}^{m \times m}\,, \tag{5.1}$$

is a function that returns a matrix $\mathbf{A}$ such that $\mathbf{A}$ is a valid covariance ($\mathbf{A} \succ 0$ and $\mathbf{A} = \mathbf{A}^\top$) and for each of its eigenvalues $e \in \mathsf{eig}(\mathbf{A})$, there exists a negligible function [Def. 3.4][katzIntroductionModernCryptography2008] η such that $e \leq \eta(\kappa)$.

We can now introduce the security notion that captures the formal requirements of the problem we want to solve.

Definition 5.1.4. A privileged estimation scheme meets notion $\{\mathbf{D}_1, \mathbf{D}_2, \dots\}$-*Covariance Privilege for Models* $\mathcal{M}_S$ *and* $\mathcal{M}_M$ if for any PPT estimator $\mathcal{A}$, there exists a PPT estimator $\mathcal{A}'$, such that

$$\begin{aligned} &\mathsf{Cov}\left[\mathcal{A}\left(k, \kappa, \mathsf{pub}, \mathcal{M}_S, \mathcal{M}_M, \underline{y}'_1, \dots, \underline{y}'_k\right) - \underline{x}_k\right] \\ &\quad - \mathsf{Cov}\left[\mathcal{A}'\left(k, \kappa, \mathsf{pub}, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1, \dots, \underline{y}_k\right) - \underline{x}_k\right] \\ &\quad\quad \succeq \mathbf{D}_k + \mathsf{neglCov}_m(\kappa) \end{aligned} \tag{5.2}$$

for valid covariances $\mathbf{D}_k$ and some negligible covariance function for all $k > 0$. Here, estimators $\mathcal{A}$ and $\mathcal{A}'$ are running in polynomial-time with respect to the security parameter κ, and all probabilities are taken over randomness introduced in models $\mathcal{M}_S$ and $\mathcal{M}_M$, estimators $\mathcal{A}$ and $\mathcal{A}'$, and algorithms Setup and Noise.

Informally, the above definition states that no estimator that can only access noisy measurements $\underline{y}'_1, \dots, \underline{y}'_k$ can estimate a state $\underline{x}_k$ for a time step k with a mean square error (MSE) covariance less than an equivalent estimator with access to true measurements $\underline{y}_1, \dots, \underline{y}_k$, by a margin of at least $\mathbf{D}_k$. We also note that by taking probabilities over randomness introduced in the system model, and therefore the possible true states $\underline{x}_k$, the definition fits a Bayesian interpretation of probability for any stochastic system model.

5.1.2. Estimation Problem

With the relevant security definitions above, we now give the specific estimation models required for our scheme. The system model we consider gives the state $\underline{x}_k \in \mathbb{R}^n$ at an integer time step k and is given by

$$\underline{x}_k = \mathbf{F}_k \underline{x}_{k-1} + \underline{w}_k \,, \tag{5.3}$$

with white noise term $\underline{w}_k \sim \mathcal{N}(\underline{0}, \mathbf{Q}_k)$ and a known non-zero covariance $\mathbf{Q}_k \in \mathbb{R}^{n \times n}$. Similarly, the measurement model gives the measurement $\underline{y}_k$ at a time step k and is given by

$$\underline{y}_k = \mathbf{H}_k \underline{x}_k + \underline{v}_k \,, \tag{5.4}$$

with white noise term $\underline{v}_k \sim \mathcal{N}(\underline{0}, \mathbf{R}_k)$ and a known non-zero covariance $\mathbf{R}_k \in \mathbb{R}^{m \times m}$.

Next, we propose a privileged estimation scheme meeting definition 5.1.4 for a derivable series of covariances when models $\mathcal{M}_S$ and $\mathcal{M}_M$ are of the form (5.3) and (5.4), respectively.

5.1.3. Multi-sensor Problem

We are interested in environments where multiple sensors are present and the fusion of their measurements can lead to better state estimation accuracy of the system they are measuring. In these environments, we want to provide levels of privilege to state estimators such that those with a higher privilege can perform better than those with a lower one while taking into consideration the estimation benefits from fusing additional measurements. We will consider linear and Gaussian models, where a state $\underline{x}_k \in \mathbb{R}^n$, at an integer timestep k, follows a system model given by

$$\underline{x}_k = \mathbf{F}_k \underline{x}_{k-1} + \underline{w}_k \,, \tag{5.5}$$

with white noise term $\underline{w}_k \sim \mathcal{N}(\underline{0}, \mathbf{Q}_k)$ and a known covariance $\mathbf{Q}_k \in \mathbb{R}^{n \times n}$. Similarly, measurements $\underline{y}_{k,i} \in \mathbb{R}^m$ from each sensor i, $1 \leq i \leq N$, follow the measurement models

$$\underline{y}_{k,i} = \mathbf{H}_{k,i} \underline{x}_k + \underline{v}_{k,i} \,, \tag{5.6}$$

with white noise term $\underline{v}_{k,i} \sim \mathcal{N}(\underline{0}, \mathbf{R}_{k,i})$ and a known covariance $\mathbf{R}_{k,i} \in \mathbb{R}^{m \times m}$. In addition to these models, we assume that all sensors i are synchronised in timesteps k to simplify later cryptographic evaluation.

Each of the sensors also holds a secret key sk_i, which can be made available to estimators of appropriate privilege. The privileges that we consider, in terms of access to keys and measurements, will be defined by sequential sensors. That is, in the presence of N sensors, we consider exactly N possible privilege levels, where each privilege $p > 0$ corresponds to holding the sequential secret keys sk_j, $1 \leq j \leq p$, while being unprivileged, $p = 0$, corresponds to holding none. We assume that estimators have access to all "privileged measurements", those from sensors whose keys they hold, and can additionally fuse "unprivileged measurements" from those whose keys they do not hold. To simplify notation, we will consider access to unprivileged measurements to be sequential as well, and can therefore capture their capabilities by letting $\mathsf{e}^{[p,q]}$ denote an estimator with privilege p and access to measurements from $q \geq p$ sensors i, $1 \leq i \leq q$.

To cryptographically guarantee the difference between estimation performances, we will use the notion of covariance privilege [risticCryptographicallyPrivilegedState2022]. As we desire better performance for privileged estimators than unprivileged ones as well as to limit the gained performance from fusing unprivileged measurements, two differences will be considered.

Performance Loss Lower Bound We want to guarantee a lower bound on the estimation performance loss of any unprivileged estimator $\mathsf{e}^{[0,N]}$ on a privilege-p estimator $\mathsf{e}^{[p,p]}$ with measurements following our scheme. Naturally, this will remain a lower bound when unprivileged estimators have access to fewer unprivileged measurements or privileged estimators have access to more.

Performance Gain Upper Bound We want to guarantee an upper bound on the estimation performance gain of any estimator $\mathsf{e}^{[p,N]}$ on a privilege-p estimator $\mathsf{e}^{[p,p]}$ with measurements following our scheme. Here, the bound remains an upper bound when fewer unprivileged measurements are fused.

The resulting scheme should be such that at least two parameters are responsible for choosing the values of these two bounds.

Remark. We stress that the two bounds that will be guaranteed only bound the performances of estimators of the specified forms. That is, nothing is said about estimators which may corrupt sensors to obtain keys beyond their privilege or additional unprivileged measurements. Bounds on leakage caused by corrupting sensors can in some cases be captured by estimators of a new form $\mathsf{e}^{[p',q']}$, but are in general beyond the scope of this work.

5.2. Privileged Estimation for Linear Systems

The key idea behind the privileged estimation scheme we propose is to add pseudorandom noise to existing measurement noise at the sensor, degrading the state estimation at estimators that cannot remove it. The added noise is a keystream generated from a secret key and can be removed from measurements by any estimator holding the same key.

To allow meeting the cryptographic notion in section 4.1.1, we focus on Gaussian and linear models, where the minimum achievable error covariance is easily computable, and add a keystream of pseudorandom Gaussian noise. The keystream and added noise are given next.

5.2.1. Gaussian Keystream

To generate pseudorandom Gaussian samples, we choose to rely on first generating a typical cryptographic pseudorandom bitstream given a secret key sk. This can be done with any cryptographic stream cipher and will reduce the security of our scheme to a single, well-studied, and replaceable component. We interpret the bitstream as sequential pseudorandom integers $q_t \in \mathbb{N}$, of a suitable size, for integer indices $t > 0$ and use them to generate a sequence of pseudorandom uniform real numbers $u_t \dot{\sim} \mathcal{U}(0, 1)$.

The conversion of q_t to u_t is cryptographically non-trivial due to the floating-point representation of u_t. Since it cannot be truly representative of the distribution $\mathcal{U}(0, 1)$, the pseudorandomness of samples is affected, and meeting the desired cryptographic notion is complicated. For now, we will assume that the uniform floating-point numbers (floats) are sufficiently close to true uniform reals, as is the current industry standard, and rely on any common method for choosing the bit size of integers q_t and the pseudorandom generation of uniforms u_t, [goualardGeneratingRandomFloatingPoint2020]. In section 5.2.4, we will state and further discuss this assumption.

Given the sufficiently uniform pseudorandom floats u_t, we are left with generating a series of pseudorandom standard normal Gaussian samples, which can be readily computed using the Box-Muller transform [paleyFourierTransformsComplex1934], by

$$z_t = \sqrt{-2\ln(u_t)}\cos(2\pi u_{t+1}) \tag{5.7}$$

and

$$z_{t+1} = \sqrt{-2\ln(u_t)}\sin(2\pi u_{t+1}), \tag{5.8}$$

obtaining two, independent, standard normal Gaussian samples from two uniform ones. This Gaussian keystream can then be used by a privileged estimation scheme to add arbitrary pseudorandom multivariate Gaussian noises.

5.2.2. Measurement Modification

To use the series of pseudorandom Gaussian samples z_t, $t > 0$, at the sensor and privileged estimator, they need to be converted to n-dimension zero-mean multivariate Gaussian samples suitable for use in the measurement model (5.4), at every time step k. As we want control over the difference in estimation error between privileged and unprivileged estimators, we do so by including the symmetric matrix parameter $\mathbf{Z} \succ 0$, in a way that added pseudorandom noise $\underline{p}_k$ at time step k is such that $\underline{p}_k \dot{\sim} \mathcal{N}(\underline{0}, \mathbf{Z})$. Given $\mathbf{Z}$, $\underline{p}_k$ is computed using the next n Gaussian keystream samples, that is $(k-1)n + 1 \leq t \leq kn$, as

$$\underline{p}_k = \mathbf{A} \cdot \begin{bmatrix} z_{(k-1)n+1} & \dots & z_{kn} \end{bmatrix}^\top, \tag{5.9}$$

for any matrix $\mathbf{A}$ such that $\mathbf{A}\mathbf{A}^\top = \mathbf{Z}$. We also note that for the correct removal of noise terms $\underline{p}_k$ by the privileged estimator, index information k is required when communication channels are lossy or have some delay. While estimation over the internet may use the indexing information already present in TCP/IP, for the remainder of the work we consider the case when all measurements arrive, in order, and neglect additional index information for the sake of simplicity.

Before estimation, we assume that the secret key sk, required for generating the Gaussian keystream in section 5.2.1, has been shared between the sensor and privileged estimator. During estimation, the sensor modifies measurements $\underline{y}_k$ by

$$\underline{y}'_k = \underline{y}_k + \underline{p}_k \,, \tag{5.10}$$

resulting in a new measurement model

$$\underline{y}'_k = \mathbf{H}_k \underline{x}_k + \underline{v}_k + \underline{p}_k \,, \tag{5.11}$$

with $\underline{v}_k \sim \mathcal{N}(\underline{0}, \mathbf{R}_k)$ and $\underline{p}_k \mathrel{\dot\sim} \mathcal{N}(\underline{0}, \mathbf{Z})$. There are now two estimation problems present for the privileged and unprivileged estimator, respectively.

Privileged estimation The estimator that holds the secret key sk can compute the Gaussian key stream z_t, $t > 0$, and therefore the added noise vectors $\underline{p}_k$ at every time step k. Computing $\underline{y}_k = \underline{y}'_k - \underline{p}_k$ given the noisy measurements results in the original measurements following measurement model (5.4) exactly.

Unprivileged estimation In the case where pseudorandomness is indistinguishable from randomness, as is the case at an unprivileged estimator when using a cryptographically secure keystream and sk is not known, noisy measurements are indistinguishable from those following the unprivileged measurement model

$$\underline{y}'_k = \mathbf{H}_k \underline{x}_k + \underline{v}'_k \,, \tag{5.12}$$

with $\underline{v}'_k \sim \mathcal{N}(\underline{0}, \mathbf{R}_k + \mathbf{Z})$, exactly.

Intuitively, we can see that the two estimators will have their difference in estimation error dependent on matrix $\mathbf{Z}$. In the security section, we will show that the best possible error covariances achievable by the privileged and unprivileged estimators can be computed exactly for both measurement models and that the difference between them will give the series $\mathbf{D}_k$, $k > 0$, required for the security notion in definition 5.1.4.

5.2.3. Multiple Privileges

In the above scenario, we have considered a single estimation privilege with one private key, dividing estimation error covariance into two groups. As a direct extension, it may be desirable to define multiple *levels* of privilege, such that the best estimation performance depends on the privilege level of an estimator. Here we will briefly put forward one such example, where a single secret key corresponds to each privilege level and noise is added similarly to (5.10) for each key individually.

We now have N secret keys sk_i and covariances for the added noises $\mathbf{Z}_i$, $1 \leq i \leq N$. Sensor measurements are modified by

$$\underline{y}''_k = \underline{y}_k + \underline{p}_k^{(1)} + \cdots + \underline{p}_k^{(N)} \,, \tag{5.13}$$

with $\underline{p}_k^{(i)} \mathrel{\dot\sim} \mathcal{N}(\underline{0}, \mathbf{Z}_i)$, $1 \leq i \leq N$. From (5.13), we see that obtaining any single key sk_i leads to a measurement model where only a single pseudorandom Gaussian sample, of covariance $\mathbf{Z}_i$, is removed, resulting in measurements indistinguishable from those following the unprivileged measurement model

$$\underline{y}_k^{(i)} = \mathbf{H}_k \underline{x}_k + \underline{v}_k^{(i)} \,, \tag{5.14}$$

where $\underline{v}_k^{(i)} \sim \mathcal{N}(\underline{0}, \mathbf{R}_k + \mathbf{E}_i)$, with

$$\mathbf{E}_i = \sum_{j=1, j\neq i}^{N} \mathbf{Z}_j \,. \tag{5.15}$$

As values $\mathbf{E}_i$ directly correspond to the relative estimation performances of each privilege level, we are also interested in the numerical restrictions when choosing these matrices. For the models to be valid for any measurement covariance $\mathbf{R}_k$, it is clear that $\mathbf{E}_i \succ 0$ and $\mathbf{E}_i = \mathbf{E}_i^\top$ must hold for all $1 \leq i \leq N$, but due to the dependence of $\mathbf{Z}_i$, there is an additional restriction required to ensure all values of $\mathbf{Z}_i$ remain valid covariances as well. From (5.15) we can write

$$\begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{I} & \cdots & \mathbf{I} \\ \mathbf{I} & \mathbf{0} & \mathbf{I} & \cdots & \mathbf{I} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \mathbf{I} & \cdots & \mathbf{I} & \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \cdots & \mathbf{I} & \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{Z}_1 \\ \mathbf{Z}_2 \\ \vdots \\ \mathbf{Z}_{N-1} \\ \mathbf{Z}_N \end{bmatrix} = \begin{bmatrix} \mathbf{E}_1 \\ \mathbf{E}_2 \\ \vdots \\ \mathbf{E}_{N-1} \\ \mathbf{E}_N \end{bmatrix} , \tag{5.16}$$

which, when rearranged and the conditions $\mathbf{Z}_i \succ 0$ are taken into account, gives the additional requirement

$$\mathbf{E}_i \prec \frac{1}{N-1} \sum_{j=1}^{N} \mathbf{E}_j \tag{5.17}$$

for all $1 \leq i \leq N$.

From (5.17) we can see that privilege levels are significantly restricted in relative estimation performance. We have demonstrated this method due to its simplicity and relation to the single level scheme, however, alternative methods involving multiple or overlapping keys may allow weaker restrictions and will be considered in future work.

5.2.4. Security Analysis

The security of the proposed scheme will be primarily considered in the single level privileged estimation case as introduced in section 5.2.2 and a proof sketch will be provided to show how the proposed scheme meets the cryptographic notion in definition 5.1.4. The extension to multiple privilege levels as described in section 5.2.3 will be informally discussed afterward.

Single Privilege

Recalling definition 5.1.4, we aim to show how the notion is met by our single privilege level estimation scheme. Before the proof sketch, we look at our scheme in the context of a formal privileged estimation scheme with model constraints and give some relevant optimality properties.

We consider the stochastic system model (5.3) and measurement model (5.4) exactly, that is, any linear models with known covariance, zero-mean, Gaussian additive noises. We define these as our model conditions and capture all relevant parameters from the respective equations in $\mathcal{M}_S$ and $\mathcal{M}_M$. Our scheme fulfills the two required algorithms for a privileged estimation scheme, Setup and Noise, as follows.

Setup Initialize a cryptographically indistinguishable stream cipher with the parameter κ, set the secret key sk to the stream cipher key and include an initial filter estimate $\hat{\underline{x}}_0$, error covariance $\mathbf{P}_0$ and added noise covariance $\mathbf{Z}$ in the public parameters pub.

Noise Computed by (5.10), returning $\underline{y}'_k$ as the noisy measurement at time step k, with added pseudorandom Gaussian noise computed from the stream cipher using sk.

Additionally, we note that in the above Setup algorithm, the inclusion of the initial state and added noise covariance are not a requirement for the security of the scheme, but merely make relevant estimation parameters public for completeness.

The idea behind our security proof relies on the optimality of the linear Kalman Filter (KF) [haugBayesianEstimationTracking2012]. Given an initial estimate and its error covariance, the KF produces posterior estimates with the minimum mean square error (MSE) achievable for *any* estimator when all measurements $\underline{y}_1, \dots, \underline{y}_k$ are observed, models are Gaussian and linear, and the same initialization is used. Since the KF also preserves initial error covariance order,

$$\mathbf{P}_k \preceq \mathbf{P}'_k \implies \mathbf{P}_{k+1} \preceq \mathbf{P}'_{k+1}\,, \tag{5.18}$$

we can define an error covariance lower-bound $\mathbf{P}_k^{(l)}$ for all possible initialisations by setting $\mathbf{P}_0^{(l)} = \mathbf{0}$ and computing the posterior KF error covariance using the combined predict and update equations

$$\begin{aligned}\mathbf{P}_k^{(l)} = &\Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top \cdot \\ &\qquad \big(\mathbf{H}_k(\mathbf{F}_k\mathbf{P}_{k-1}^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top + \mathbf{R}_k\big)^{-1}\mathbf{H}_k\Big) \cdot \\ &\Big(\mathbf{F}_k\mathbf{P}_{k-1}^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k\Big)\,.\end{aligned} \tag{5.19}$$

This gives us a lower-bound at every time step k, such that

$$\mathbf{P}_k^{(l)} \preceq \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1, \dots, \underline{y}_k\right) - \underline{x}_k\right] \tag{5.20}$$

for any estimator $\mathcal{A}$ following definition 5.1.2 and any Gaussian and linear models $\mathcal{M}_S$ and $\mathcal{M}_M$. This leads us to the security proof sketch.

Theorem 5.2.1. Our single privilege estimation scheme in section 5.2.2 meets $\{\mathbf{D}_1, \mathbf{D}_2, \dots\}$-Covariance Privilege for Models $\mathcal{M}_S$ and $\mathcal{M}_M$, for a computable series $\mathbf{D}_k$, $k > 0$ dependent on a noise parameter $\mathbf{Z}$, when $\mathcal{M}_S$ and $\mathcal{M}_M$ are Gaussian and linear.

Proof Sketch

Since a cryptographically pseudorandom stream cipher is used in section 5.2.1, the stream integers q_t, and therefore the uniform samples u_t and normal Gaussian samples z_t, are indistinguishable to those generated from a truly random stream for any PPT estimator without the secret key. We persist with the previous assumption that floating-point representations of z_t are sufficiently close to Gaussian and assume the KF to provide optimal estimation when using floats, as is standard in the state-of-the-art. Using the Setup and Noise algorithms for our scheme now leads to pseudorandom noisy measurements $\underline{y}'_k$ that are indistinguishable from measurements following the unprivileged measurement model (5.12). We can now compute a lower-bound $\mathbf{P}'^{(l)}_k$ for any unprivileged estimator as $\mathbf{P}'^{(l)}_0 = \mathbf{0}$ and

$$
\begin{aligned}
\mathbf{P}'^{(l)}_k = &\Big(\mathbf{I} - (\mathbf{F}_k \mathbf{P}'^{(l)}_{k-1} \mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top \cdot \\
&\quad \big(\mathbf{H}_k (\mathbf{F}_k \mathbf{P}'^{(l)}_{k-1} \mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top + \mathbf{R}_k + \mathbf{Z}\big)^{-1} \mathbf{H}_k\Big) \cdot \\
&\Big(\mathbf{F}_k \mathbf{P}'^{(l)}_{k-1} \mathbf{F}_k^\top + \mathbf{Q}_k\Big) .
\end{aligned}
\tag{5.21}
$$

Taking the difference of (5.21) and (5.19) produces the series

$$
\mathbf{D}_k = \mathbf{P}'^{(l)}_k - \mathbf{P}^{(l)}_k , \tag{5.22}
$$

for $k > 0$, which can be tuned by the parameter $\mathbf{Z}$. Since both series $\mathbf{P}^{(l)}_k$ and $\mathbf{P}'^{(l)}_k$ give the lowest possible error covariance of the respective estimators, an estimator knowing model (5.4) can always be created for one knowing only model (5.12) such that their error covariances at any time step k differ by at least $\mathbf{D}_k$. A reduction proof can be easily constructed, where the existence of an unprivileged estimator in our scheme, that can produce estimates such that (5.2) does not hold, can be used to construct an estimator with an error covariance lower than $\mathbf{P}'^{(l)}_k$ given a known model of the form (5.12). As no such estimator exists, we conclude that our scheme meets $\{\mathbf{D}_1, \mathbf{D}_2, \dots\}$-Covariance Privilege for Models $\mathcal{M}_S$ and $\mathcal{M}_M$, when $\mathcal{M}_S$ and $\mathcal{M}_M$ are Gaussian and linear. This concludes our proof sketch.

In addition to the proof sketch, we stress caution when accepting a cryptographic guarantee in terms of models $\mathcal{M}_S$ and $\mathcal{M}_M$ when used to estimate a measured physical process or approximate continuous quantities. The following assumptions are made in this scenario.

Exact models When assigning a model to a physical process, any cryptographic guarantees concerning the model assume the process follows the model *exactly*. It is often the case that models assume a Bayesian interpretation of probability (a stochastic state) or are chosen to simplify estimation, resulting in the possibility of better estimation given alternative or more complicated models. Although the standard for state estimation, we state the assumption to highlight the distinction between models and a physical process.

Floating-point approximation As stated in section 5.2.1 and the proof sketch above, floating-point approximations to real numbers complicate cryptographic guarantees when relying on proofs using real numbers such as KF optimality. While optimal estimation with floats is beyond the scope of this work, the prevalence of floats in decades of state estimation justifies the assumption of sufficient similarity and the insignificance of associated error introduced to the security notion.

Multiple Additional Noises

We have not defined a security notion for multiple levels of privileged estimation from section 5.2.3, but an intuitive and informal extension is briefly described here.

A suitable notion would require that for any subset of corrupted estimators, and thus estimators with any subset of secret keys $S \subseteq \{\mathsf{sk}_i, 1 \leq i \leq N\}$, who are given noisy measurements $\underline{y}''_k$, an estimator given true measurements $\underline{y}_k$ can be constructed such that the difference between the corrupted subset's error covariance and its own is at least $\mathbf{D}_k^{(S)}$ at time step k. Although this definition requires a series $\mathbf{D}_k^{(S)}$ for every possible subset of keys, S, complicating its formal specification, it captures the exact advantage of every such subset producing a general definition.

Given the structure of our scheme in section 5.2.3, it can be readily seen how the above notion would be met. Similarly to the single level case, the KF can be used to compute the minimum error covariances for all compromised key subsets as well as for an estimator with the true measurements, and the relevant difference series $\mathbf{D}_k^{(S)}$ can be defined.

Non-Linear Systems

5.2.5. Simulation

As well as showing the theoretical security of our scheme, we have simulated the stochastic estimation problem using linear Kalman filter estimators for the different measurement models. Simulations have been implemented in the Python language and use the AES block cipher in CTR mode as a cryptographically secure stream cipher (AES-CTR) [gueronIntelAdvancedEncryption2010].

We consider two simulations, both following the same two-dimensional time-invariant constant velocity system model, given by

$$\mathbf{F}_k = \begin{bmatrix} 1 & 0.5 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \ \mathbf{Q}_k = \frac{1}{10^3} \cdot \begin{bmatrix} 0.4 & 1.3 & 0 & 0 \\ 1.3 & 5.0 & 0 & 0 \\ 0 & 0 & 0.4 & 1.3 \\ 0 & 0 & 1.3 & 5.0 \end{bmatrix},$$

for all k, with differing measurement models. In all cases, estimators were initialized with the same initial conditions, equal to the true starting condition of the system they were estimating, with initial error covariance $\mathbf{0}$.

The first measurement model measures location and leads to an observable system with bounded error covariances as $k \to \infty$. It is given by $\mathbf{H}_k = \left[\begin{smallmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{smallmatrix}\right]$ and $\mathbf{R}_k = \left[\begin{smallmatrix} 5 & 2 \\ 2 & 5 \end{smallmatrix}\right]$, for all k, and the sensor adds pseudorandom Gaussian samples with covariance $\mathbf{Z} = 35 \cdot \mathbf{I}$ to create an estimator privilege level. Figure 5.1 shows the average error covariance traces and the mean square error (MSE) of estimation from 1000 runs of our privileged estimation scheme, where the above models are followed. It can be seen that the trace of the privileged estimator's error covariance stays lower than that of the unprivileged one and that privileged estimation has lower MSE. The difference in trace between the two estimators has also been plotted and is equal to the trace of the series (5.22) for all time steps k due to the chosen initial error covariance.

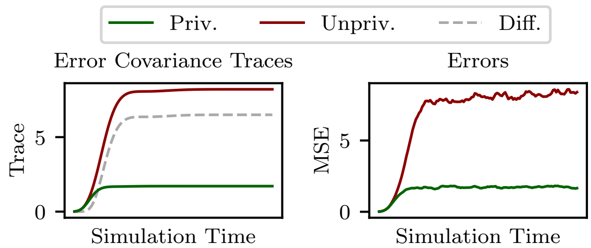


Figure 5.1.: Privileged estimation with bounded error covariance.

The second simulation considers an unobservable system where only the velocity is measured and has an unbounded error covariance as $k \to \infty$. It is given by $\mathbf{H}_k = \left[\begin{smallmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{smallmatrix}\right]$, for all k, and uses the same values for $\mathbf{R}_k$ and $\mathbf{Z}$ as the previous model. Figure 5.2 shows the average error covariance traces and MSE of estimation from 1000 runs using this model and captures how error covariance boundedness does not affect the privileged estimation scheme's properties.

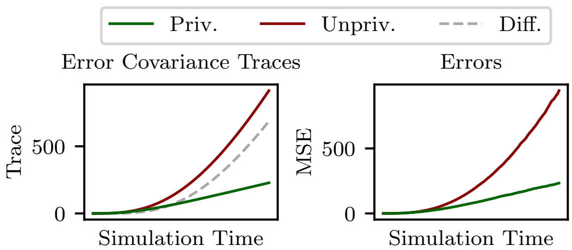


Figure 5.2.: Privileged estimation with unbounded error covariance.

Lastly, a simulation of multiple privilege levels was also performed using the bounded error covariance measurement model and using pseudorandom Gaussian samples such that $\mathbf{E}_1 = 20 \cdot \mathbf{I}$, $\mathbf{E}_2 = 14 \cdot \mathbf{I}$, and $\mathbf{E}_3 = 17 \cdot \mathbf{I}$ for estimators holding the single keys sk_1, sk_2 and sk_3, respectively. Note that the three matrices $\mathbf{E}_i$, $1 \le i \le 3$ satisfy (5.17). Figure

5.3 again shows the average traces and MSE of estimation from 1000 runs and displays the distinct difference in estimation error of the different privilege levels. Additionally, two special cases that bound all estimators are included, one holding all privilege level keys and another holding none.

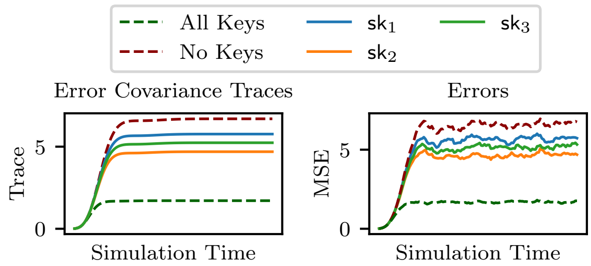


Figure 5.3.: Estimation with multiple privilege levels.

All of the included figures capture the difference in estimation error between the best possible estimators given the simulated processes (in terms of MSE) and support the proposed security proof sketch given in section 5.2.4.

5.3. Fusion in Privileged Estimation Environments

The idea behind our privileged fusion scheme is to add *correlated* Gaussian keystreams to the measurements from each sensor. These noises can be computed and subtracted by estimators holding respective sensor keys, while their correlation limits the additional information gained from fusing unprivileged measurements.

5.3.1. Noise Generation

Similarly to the Gaussian keystream generation in [eqn:gaussian_keystream_generation], pseudorandom samples can be correlated in this way even when generated using different stream cipher keys. To parameterise the correlation between noises at each sensor, we introduce a fully correlated component $\mathbf{Z} \in \mathbb{R}^{m \times m}$ and an uncorrelated component $\mathbf{Y} \in \mathbb{R}^{m \times m}$ and define a noise cross-correlation matrix for x noises as $\mathbf{S}^{(x)} \in \mathbb{R}^{xm \times xm}$,

$$\mathbf{S}^{(x)} = \begin{bmatrix} \mathbf{Z} & \cdots & \mathbf{Z} \\ \vdots & \ddots & \vdots \\ \mathbf{Z} & \cdots & \mathbf{Z} \end{bmatrix} + \begin{bmatrix} \mathbf{Y} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Y} \end{bmatrix}, \tag{5.23}$$

and $\mathbf{S}^{(1)} = \mathbf{Z} + \mathbf{Y}$. The generation of all N multivariate Gaussian noises at timestep k, $\underline{g}_k^{(1:N)}$, can now be written as

$$\underline{g}_k^{(1:N)} = \begin{bmatrix} \underline{g}_{k,1} \\ \vdots \\ \underline{g}_{k,N} \end{bmatrix} = \mathbf{S}^{(N)\frac{1}{2}} \cdot \begin{bmatrix} \underline{z}_{k,1} \\ \vdots \\ \underline{z}_{k,N} \end{bmatrix} , \tag{5.24}$$

where each $\underline{z}_{k,i}$ is computed as $\underline{z}_k$ in [eqn:standard_gaussian_generation] using uniform samples generated with key sk_i, and $\mathbf{S}^{(N)\frac{1}{2}}$ is a matrix such that $\mathbf{S}^{(N)\frac{1}{2}}\mathbf{S}^{(N)\frac{1}{2}\top} = \mathbf{S}^{(N)}$. Notably, it is important that the vector of the first p noises $\underline{g}_{k,i}$, $1 \leq i \leq p$, in (5.24), denoted $\underline{g}_k^{(1:p)}$, can be reproduced by an estimator of privilege p, holding only the keys sk_i, $1 \leq i \leq p$. One case where this is possible is when a lower-triangular decomposition, such as the Cholesky decomposition, is used to compute $\mathbf{S}^{(N)\frac{1}{2}}$ from $\mathbf{S}^{(N)}$. Here, each correlated Gaussian sample $\underline{g}_{k,i}$ is computable from preceeding uniform samples $\underline{z}_{k,j}$, $j \leq i$ only, and the generalised noise generation equation

$$\underline{g}_k^{(1:p)} = \mathbf{S}^{(p)\frac{1}{2}} \cdot \begin{bmatrix} \underline{z}_{k,1} \\ \vdots \\ \underline{z}_{k,p} \end{bmatrix} \tag{5.25}$$

generates the same first p noises $\underline{g}_k^{(1:p)}$ as would be obtained from (5.24). This is due to $\mathbf{S}^{(p)\frac{1}{2}} \in \mathbb{R}^{pm \times pm}$ equalling the top left block of matrix $\mathbf{S}^{(N)\frac{1}{2}}$ when using a lower-triangular decomposition.

With (5.25), at every timestep k, $\underline{g}_k^{(1:N)}$ can be generated using all N keys and used to modify sensor measurements, while the subset $\underline{g}_k^{(1:p)}$ can be generated by estimators of privilege p using only the keys they hold.

5.3.2. Measurement Modification

With a way to generate noises for sensors and estimators, we can introduce the means of measurement modification and the observable measurement models for different estimators. Measurement modification is peformed by adding noises $\underline{g}_k^{(1:N)}$ to measurements from each sensor i before making them public, resulting in modified measurement equations for each sensor,

$$\begin{aligned} \underline{y}'_{k,i} &= \underline{y}_{k,i} + \underline{g}_{k,i} \\ &= \mathbf{H}_{k,i}\underline{x}_k + \underline{v}_{k,i} + \underline{g}_{k,i} , \end{aligned} \tag{5.26}$$

with real measurement noise $\underline{v}_{k,i} \sim \mathcal{N}(\underline{0}, \mathbf{R}_{k,i})$ and the vector of all added noises $\underline{g}_k^{(1:N)} \dot{\sim} \mathcal{N}(\underline{0}, \mathbf{S}^{(N)})$. As we assume that sensors are synchronised in k, we can capture the correlation between these modified measurements exactly by considering the stacked measurement model for any estimator with access to q measurements, $\mathsf{e}^{[p,q]}$, at time k, given by

$$\begin{aligned}\underline{y}_k'^{(1:q)} &= \underline{y}_k^{(1:q)} + \underline{g}_k^{(1:q)} \\ &= \mathbf{H}_k^{(1:q)} \underline{x}_k + \underline{v}_k^{(1:q)} + \underline{g}_k^{(1:q)}\end{aligned} \tag{5.27}$$

where $\underline{v}_k^{(1:q)} \sim \mathcal{N}(\underline{0}, \mathbf{R}_k^{(1:q)})$ and $\underline{g}_k^{(1:q)} \dot\sim \mathcal{N}(\underline{0}, \mathbf{S}^{(q)})$, with

$$\underline{y}_k'^{(1:q)} = \begin{bmatrix} \underline{y}_{k,1}' \\ \vdots \\ \underline{y}_{k,q}' \end{bmatrix}, \ \underline{y}_k^{(1:q)} = \begin{bmatrix} \underline{y}_{k,1} \\ \vdots \\ \underline{y}_{k,q} \end{bmatrix}, \ \mathbf{H}_k^{(1:q)} = \begin{bmatrix} \mathbf{H}_{k,1} \\ \vdots \\ \mathbf{H}_{k,q} \end{bmatrix},$$

$$\underline{v}_k^{(1:q)} = \begin{bmatrix} \underline{v}_{k,1} \\ \vdots \\ \underline{v}_{k,q} \end{bmatrix}, \ \mathbf{R}_k^{(1:q)} = \begin{bmatrix} \mathbf{R}_{k,1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_{k,q} \end{bmatrix}$$

and $\mathbf{S}^{(q)} \in \mathbb{R}^{qm \times qm}$ defined by (5.23).

Since we are using a cryptographically sound stream cipher to generate the added Gaussian keystream, the pseudorandom samples are indistinguishable from truly random ones to estimators without appropriate keys, which leads us to three observable measurement models, *i.e.*, the models that capture all the information available to an estimator exactly, for three types of mutually exhaustive estimators.

Estimators of the form $\mathsf{e}^{[0,q]}$ Here, no keys are held by an unprivileged estimator with access to q measurements, thus all generated noises $\underline{g}_k^{(1:q)}$ are indistinguishable from noises from the truly random distribution $\mathcal{N}(\underline{0}, \mathbf{S}^{(q)})$. For these estimators, we can rewrite the measurement equation (5.27) as the observed measurement model

$$\underline{y}_k^{[0,q]} = \mathbf{H}_k^{(1:q)} \underline{x}_k + \underline{v}_k', \tag{5.28}$$

with truly Gaussian term $\underline{v}_k' \sim \mathcal{N}(\underline{0}, \mathbf{R}_k^{(1:q)} + \mathbf{S}^{(q)})$.

Estimators of the form $\mathsf{e}^{[p,p]}$ Estimators with keys for all the sensors to which they have access can generate all added noises and subtract them from the received measurements. That is, $\underline{g}_k^{(1:p)}$ can be generated and $\underline{y}_k^{[p,p]} = \underline{y}_k'^{(1:p)} - \underline{g}_k^{(1:p)}$ computed to give the observed measurement model equal to recieving unmodified measurements only,

$$\underline{y}_k^{[p,p]} = \mathbf{H}_k^{(1:p)} \underline{x}_k + \underline{v}_k^{(1:p)}, \tag{5.29}$$

where $\underline{v}_k^{(1:p)} \sim \mathcal{N}(\underline{0}, \mathbf{R}_k^{(1:p)})$.

Estimators of the form $\mathsf{e}^{[p,q]}$, $p < q$ Lastly, we want the observed measurement model when only some accessible measurements can have their noises removed. Here, the noises from sensors $i > p$ which cannot be removed are conditionally dependent on the known noises $\underline{g}_k^{(1:p)}$. Since we can generate the noises $\underline{g}_k^{(1:p)}$ and know that $\underline{g}_k^{(1:q)} \dot{\sim} \mathcal{N}(\underline{0}, \mathbf{S}^{(q)})$, we can write

$$\begin{aligned}\underline{g}_k^{(1:q)} &= \begin{bmatrix} \underline{g}_k^{(1:p)} \\ \underline{g}_k^{(p+1:q)} \end{bmatrix} \\ &\dot{\sim} \mathcal{N}\left(\begin{bmatrix} \underline{0} \\ \underline{0} \end{bmatrix}, \begin{bmatrix} \mathbf{S}^{(p)} & \bar{\mathbf{Z}} \\ \bar{\mathbf{Z}}^\top & \mathbf{S}^{(q-p)} \end{bmatrix}\right),\end{aligned} \tag{5.30}$$

where $\bar{\mathbf{Z}} \in \mathbb{R}^{pm \times (q-p)m}$ is a block matrix with every block equal to $\mathbf{Z}$, and compute the conditional pseudorandom Gaussian distribution

$$\begin{aligned}&\underline{g}_k^{(p+1:q)} \mid \underline{g}_k^{(1:p)} \\ &\quad \dot{\sim} \mathcal{N}\Big(\bar{\mathbf{Z}}^\top \mathbf{S}^{(p)-1} \underline{g}_k^{(1:p)}, \\ &\qquad\qquad \mathbf{S}^{(q-p)} - \bar{\mathbf{Z}}^\top \mathbf{S}^{(p)-1} \bar{\mathbf{Z}}\Big).\end{aligned} \tag{5.31}$$

Now, subtracting the known noises $\underline{g}_k^{(1:p)}$ and the means of the unknown noises (5.31) from recieved measurements,

$$\underline{y}_k^{[p,q]} = \underline{y}_k'^{(1:q)} - \begin{bmatrix} \underline{g}_k^{(1:p)} \\ \bar{\mathbf{Z}}^\top \mathbf{S}^{(p)-1} \underline{g}_k^{(1:p)} \end{bmatrix}, \tag{5.32}$$

and accounting for unknown pseudorandom noises being indistinguishable from random, a zero-mean observed measurement model can be written as

$$\underline{y}_k^{[p,q]} = \mathbf{H}_k^{(1:q)} \underline{x}_k + \underline{v}_k' \tag{5.33}$$

where

$$\begin{aligned}\underline{v}_k' \sim \mathcal{N}\Bigg(&\underline{0}, \\ &\begin{bmatrix} \mathbf{R}_k^{(1:p)} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}^{(q-p)} - \bar{\mathbf{Z}}^\top \mathbf{S}^{(p)-1} \bar{\mathbf{Z}} + \mathbf{R}_k^{(p+1:q)} \end{bmatrix}\Bigg).\end{aligned}$$

Remark. Recalling that we assume estimators access unprivileged measurements sequentially to simplify notation, (5.30), (5.32) and (5.33) can be generalised when having access to arbitrary $q - p$ non-sequential unprivileged measurements $\underline{y}_{k,i}$, $p < i \leq q$, by appropriately rearranging the columns of $\mathbf{S}^{(q-p)}$ in (5.30).

From the observed measurement models (5.28), (5.29) and (5.33) we can tell that the parameters $\mathbf{Z}$ and $\mathbf{Y}$ (within matrices $\mathbf{S}^{(q)}$, $\mathbf{S}^{(p)}$ and $\mathbf{S}^{(q-p)}$) will control the difference in estimation performance between the three types of estimators. Computing the two differences we wish to cryptographically guarantee from section [sec:prob], and how $\mathbf{Z}$ and $\mathbf{Y}$ affect them, will be more formally explored in sections 5.3.4 and 5.3.5.

5.3.3. Distribution of Noise Terms

While we have described a method for generating noises that modify N measurements and result in different observed measurement models depending on estimator privilege, we have not discussed where the noise is generated and how it is distributed to sensors. To handle the inherent correlation of the noises $\underline{g}_k^{(1:N)}$, they can be generated either centrally before distribution to sensors or sequentially at the sensors themselves, given previously generated values.

Central noise generation To compute noises centrally, (5.25) can be computed for all N noises at a central processor and each noise $\underline{g}_{k,i}$ sent to the respective sensor i before it modifies its local measurement by (5.26).

Sequential noise generation To compute the same noises sequentially for each timestep k, sensor 1 can generate its noise independently using its current standard Gaussian sample $\underline{z}_{k,1}$, by $\underline{g}_{k,1} = \mathbf{S}^{(1)\frac{1}{2}}\underline{z}_{k,1}$. Each following sensor $i > 1$ can generate its noise $\underline{g}_{k,i}$ given the preceeding noises $\underline{g}_k^{(1:i-1)}$ and following the conditional reasoning in (5.31), as

$$\begin{aligned}\underline{g}_{k,i} =& \bar{\mathbf{Z}}^\top \mathbf{S}^{(i-1)-1}\underline{g}_k^{(1:i-1)} + \\ & (\mathbf{S}^{(1)} - \bar{\mathbf{Z}}^\top \mathbf{S}^{(i-1)-1}\bar{\mathbf{Z}})^{\frac{1}{2}}\underline{z}_{k,i}\,.\end{aligned} \tag{5.34}$$

After local noise generation, sensor i sends its and preceding noises, $\underline{g}_k^{(1:i)}$, to the next sensor $i+1$. This method has the clear downside of increasing communication costs with each successive generation but requires no central communicator.

In both cases above, the computation of all noises $\underline{g}_k^{(1:N)}$ can be performed offline, reducing the complexity of real-time measurement modification.

5.3.4. Security Analysis

To give proof sketches of the cryptographic guarantees provided by the presented scheme, we first recall some assumptions. We consider floating-point numbers to be sufficiently close to real random numbers that real-number proofs still hold, and that all sensors are synchronised in k such that observed measurement models (5.28), (5.29) and (5.33) are exactly correct. Using these assumptions, the proofs rely on the optimality of the linear Kalman filter (KF) [haugBayesianEstimationTracking2012] to produce a series of covariances for optimal estimators before taking their differences to obtain the *Performance Loss Lower Bound* and *Performance Gain Upper Bound* from section [sec:prob]. Similarly to [risticCryptographicallyPrivilegedState2022], these series can be used as

$\mathbf{D}_1, \mathbf{D}_2, \ldots$ in definition [def:cov_priv_security_notion] for appropriately formulated privileged estimation schemes for the two bounds, and demonstrate that the existence of an estimator violating the notion implies the existence of a linear estimator with error covariance lower than the KF. This guarantees the bounds by contrapositive.

Performance Loss Lower Bound (PLLB)

First, we consider the lower bound to the loss in estimation performance an estimator $\mathsf{e}^{[0,N]}$ has on an estimator $\mathsf{e}^{[p,p]}$ when measurements follow the presented scheme. Since the observed measurement models for these estimators, (5.28) and (5.29), interpret available measurements as a single stacked measurement, and since we do not consider estimators that corrupt sensors, we can treat the stacked measurement as coming from a single sensor and use the notion of covariance privilege in definition [def:cov_priv_security_notion] to guarantee the bound. The associated privileged estimation scheme for the PLLB can be written for each privilege p as

Setup Given the system model (5.5), all measurements models (5.6) (interpretable as a single stacked measurement model) and a security parameter κ used by all sensors, generate N stream cipher keys $\mathsf{sk_i}$, $1 \leq i \leq N$, and let the secret key sk include all N keys. Generate the correlated and uncorrelated noise components $\mathbf{Z}$ and $\mathbf{Y}$, an initial estimate and error covariance $\underline{\hat{x}}_0$ and $\mathbf{P}_0$, and include these in the public parameters pub.

$\mathsf{Noise}_{\mathsf{PLLB}}$ Given parameters, cipher keys, a timestep k and true sensor measurements $\underline{y}_k^{(1:N)}$, let $\underline{y}_k^{\{\mathsf{up}\}} = \underline{y}_k^{[0,N]}$ following (5.28) and $\underline{y}_k^{\{\mathsf{p}\}} = \underline{y}_k^{[p,p]}$ following (5.29).

With the above formulation, we can use the KF to recursively compute the optimal estimate error covariances for estimators with access to only measurements $\underline{y}_k^{\{\mathsf{up}\}}$ or $\underline{y}_k^{\{\mathsf{p}\}}$, for all k. Since the KF preserves initial covariance order, $\mathbf{P}_k \succeq \mathbf{P}'_k \implies \mathbf{P}_{k+1} \succeq \mathbf{P}'_{k+1}$, a lower bound can be guaranteed when using the initial covariance $\mathbf{P}_0 = \mathbf{0}$. Therefore, the minimum achievable error covariance for an estimator $\mathsf{e}^{[0,N]}$, with access to measurements $\underline{y}_k^{\{\mathsf{up}\}} = \underline{y}_k^{[0,N]}$, is given by the combined KF predict and update equations

$$\begin{aligned}\mathbf{P}_k^{[0,N]} = &\Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}^{[0,N]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top} \cdot \\ &\quad (\mathbf{H}_k^{(1:N)}(\mathbf{F}_k\mathbf{P}_{k-1}^{[0,N]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top} \\ &\quad + \mathbf{R}_k^{(1:N)} + \mathbf{S}^{(N)}\big)^{-1}\mathbf{H}_k^{(1:N)}\Big) \cdot \\ &\Big(\mathbf{F}_k\mathbf{P}_{k-1}^{[0,N]}\mathbf{F}_k^\top + \mathbf{Q}_k\Big),\end{aligned} \tag{5.35}$$

when $\mathbf{P}_0^{[0,N]} = \mathbf{0}$. Similarly, the same can be done for an estimator $\mathsf{e}^{[p,p]}$, with access to measurements $\underline{y}_k^{\{\mathsf{p}\}} = \underline{y}_k^{[p,p]}$, as

$$\begin{aligned}\mathbf{P}_k^{[p,p]} =& \Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}^{[p,p]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:p)\top}\cdot \\ & \quad (\mathbf{H}_k^{(1:p)}(\mathbf{F}_k\mathbf{P}_{k-1}^{[p,p]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:p)\top} \\ & \quad + \mathbf{R}_k^{(1:p)})^{-1}\mathbf{H}_k^{(1:p)}\Big)\cdot \\ & \Big(\mathbf{F}_k\mathbf{P}_{k-1}^{[p,p]}\mathbf{F}_k^\top + \mathbf{Q}_k\Big)\end{aligned} \tag{5.36}$$

and $\mathbf{P}_0^{[p,p]} = \mathbf{0}$. The bounds (5.35) and (5.36) are constructed such that at every timestep k,

$$\begin{aligned}\mathbf{P}_k^{[0,N]} \preceq & \\ & \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1^{[0,N]}, \dots, \underline{y}_k^{[0,N]}\right) - \underline{x}_k\right]\end{aligned} \tag{5.37}$$

and

$$\begin{aligned}\mathbf{P}_k^{[p,p]} \preceq & \\ & \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1^{[p,p]}, \dots, \underline{y}_k^{[p,p]}\right) - \underline{x}_k\right]\end{aligned} \tag{5.38}$$

hold. Recalling definition [def:cov_priv_security_notion], and knowing that the minimum achievable covariance of the estimators is produced using the KF, it can be seen that the difference

$$\mathbf{D}_{\mathsf{PLLB},k} = \mathbf{P}_k^{[0,N]} - \mathbf{P}_k^{[p,p]} \tag{5.39}$$

produces a series where for any PPT estimator $\mathsf{e}^{[0,N]}$, an equivalent PPT estimator $\mathsf{e}^{[p,p]}$, lower bounded in error by (5.36), can always be created such that the difference between their error covariances at time k is at least $\mathbf{D}_{\mathsf{PLLB},k}$. Here, the PPT requirement guarantees the indistinguishabilty of pseudorandom streams to truly random ones and a negligible performance gain for $\mathsf{e}^{[0,N]}$ is present on average if secret keys or keystreams are guessed. The existence of an estimator of the form $\mathsf{e}^{[0,N]}$ where this condition cannot be met, implies the existence of an estimator with error covariances smaller than the KF for linear models. As no such estimator exists, we conclude that the Setup and $\mathsf{Noise}_{\mathsf{PLLB}}$ algorithms above meet $\{\mathbf{D}_{\mathsf{PLLB},1}, \mathbf{D}_{\mathsf{PLLB},2}, \dots\}$-*Covariance Privilege for System Model* (5.5) *and Stacked Measurement Models* (5.6).

In the above, we lower bound the estimation performance loss an estimator $\mathsf{e}^{[0,N]}$ has on estimators $\mathsf{e}^{[p,p]}$. In the cases where the unprivileged estimator has access to fewer measurements, $\mathsf{e}^{[0,q]}$, $q < N$, or the privileged one to more, $\mathsf{e}^{[p,q]}$, $q > p$, the achievable difference can only increase (fewer measurements can only increase error covariance while more can only decrease it). This ensures the computed bound remains a lower bound for *any* unprivileged estimator.

Performance Gain Upper Bound (PGUB)

Similar to the lower bound above, we can use the same properties of the KF to give an upper bound to the gain in estimation performance an estimator $\mathsf{e}^{[p,N]}$ has on an estimator $\mathsf{e}^{[p,p]}$ when measurements follow the presented scheme. The associated privileged estimation scheme for the PGUB for each privilege p is given by the same Setup algorithm as in section 5.3.4 and

$\mathsf{Noise}_{\mathsf{PGUB}}$ Given parameters, cipher keys, a timestep k and true sensor measurements $\underline{y}_k^{(1:N)}$, let $\underline{y}_k^{\{\mathsf{up}\}} = \underline{y}_k^{[p,N]}$ following (5.33) and $\underline{y}_k^{\{\mathsf{p}\}} = \underline{y}_k^{[p,p]}$ following (5.29).

The minimum error covariances achievable by an estimator $\mathsf{e}^{[p,p]}$ is again given by (5.36) and $\mathbf{P}_0^{[p,p]} = \mathbf{0}$. For an estimator $\mathsf{e}^{[p,N]}$ with access to measurements $\underline{y}_k^{\{\mathsf{up}\}} = \underline{y}_k^{[p,N]}$ it is given by

$$\begin{aligned}\mathbf{P}_k^{[p,N]} =& \Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}^{[p,N]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top} \cdot \\ & \big(\mathbf{H}_k^{(1:N)}(\mathbf{F}_k\mathbf{P}_{k-1}^{[p,N]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top} \\ & + \mathbf{X}\big)^{-1}\mathbf{H}_k^{(1:N)}\Big)\Big(\mathbf{F}_k\mathbf{P}_{k-1}^{[p,N]}\mathbf{F}_k^\top + \mathbf{Q}_k\Big),\end{aligned} \tag{5.40}$$

where

$$\mathbf{X} = \begin{bmatrix} \mathbf{R}_k^{(1:p)} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}^{(N-p)} - \bar{\mathbf{Z}}^\top\mathbf{S}^{(p)-1}\bar{\mathbf{Z}} + \mathbf{R}_k^{(p+1:N)} \end{bmatrix} \tag{5.41}$$

and $\mathbf{P}_0^{[p,N]} = \mathbf{0}$. Again, the bounding series are such that (5.38) and

$$\begin{aligned}&\mathbf{P}_k^{[p,N]} \preceq \\ &\quad \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1^{[p,N]}, \dots, \underline{y}_k^{[p,N]}\right) - \underline{x}_k\right]\end{aligned} \tag{5.42}$$

hold. Now, the difference

$$\mathbf{D}_{\mathsf{PGUB},k} = \mathbf{P}_k^{[p,N]} - \mathbf{P}_k^{[p,p]} \tag{5.43}$$

produces a series where for any PPT estimator $\mathsf{e}^{[p,N]}$, an equivalent PPT estimator $\mathsf{e}^{[p,p]}$, lower bounded in error by (5.36), can always be created such that the difference between their error covariances at time k is at least $\mathbf{D}_{\mathsf{PGUB},k}$. With the same reasoning as for the lower bound, we conclude that the Setup and $\mathsf{Noise}_{\mathsf{PGUB}}$ algorithms above meet $\{\mathbf{D}_{\mathsf{PGUB},1}, \mathbf{D}_{\mathsf{PGUB},2}, \dots\}$-*Covariance Privilege for System Model* (5.5) *and Stacked Measurement Models* (5.6).

In (5.43), $\mathbf{D}_{\mathsf{PGUB},k} \preceq 0$ for all $k > 0$ and lower bounds the (negative) loss in performance an estimator $\mathsf{e}^{[p,N]}$ has on estimators $\mathsf{e}^{[p,p]}$. We refer to the bound as an upper bound as its negation $-\mathbf{D}_{\mathsf{PGUB},k}$, $k > 0$, upper bounds the estimation performance gain achievable by $\mathsf{e}^{[p,N]}$ on the estimators $\mathsf{e}^{[p,p]}$, as desired in section [sec:prob]. In the case where fewer unprivileged measurements are accessible, $\mathsf{e}^{[p,q]}$, $q < N$, this gain can only decrease, keeping the upper bound valid for any estimators $\mathsf{e}^{[p,q]}$, $q > p$.

Non-Linear Systems

5.3.5. Simulation

In addition to showing how the estimation performance loss and gain bounds can be computed, we have simulated optimal estimators to demonstrate the effects of the correlated and uncorrelated components, $\mathbf{Z}$ and $\mathbf{Y}$, respectively. The state of an aircraft $\begin{bmatrix} x & y & v_x & v_y \end{bmatrix}^\top$, capturing its position x, y (m) and velocity v_x, v_y (m/s), was simulated following a constant velocity system model given by (5.5) with parameters

$$\mathbf{F}_k = \begin{bmatrix} 1 & 0.5 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.44}$$

and

$$\mathbf{Q}_k = 10^{-3} \cdot \begin{bmatrix} 0.42 & 1.25 & 0 & 0 \\ 1.25 & 5 & 0 & 0 \\ 0 & 0 & 0.42 & 1.25 \\ 0 & 0 & 1.25 & 5 \end{bmatrix}, \tag{5.45}$$

for all k, and measured independently by location sensors i, $1 \leq i \leq N = 4$, following (5.6) with constant parameters

$$\mathbf{H}_{k,i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \text{ and } \mathbf{R}_{k,i} = \begin{bmatrix} 5 & 2 \\ 2 & 5 \end{bmatrix}, \tag{5.46}$$

for all k. Simulations were implemented in the Python programming language and the considered correlated and uncorrelated parameters were restricted to the forms $\mathbf{Z} = \Sigma_z \cdot \mathbf{I}$ and $\mathbf{Y} = \Sigma_y \cdot \mathbf{I}$ for simplicity. All estimators executed linear Kalman filters with the parameters above and the exact knowledge of the initial state ($\mathbf{P}_0 = \mathbf{0}$).

Figure 5.4 shows the errors of different privileged estimators with access to varying sensor measurements when added noise parameters Σ_z and Σ_y are held constant. As would be expected, the error decreases when more keys are available, while a further decrease is achieved as more additional unprivileged measurements are fused. Here, the difference in mean squared error (MSE) between $\mathsf{e}^{[0,4]}$ and $\mathsf{e}^{[p,p]}$ (shaded blue region), and between $\mathsf{e}^{[p,p]}$ and $\mathsf{e}^{[p,4]}$ (shaded red region), are bounded on average by the trace of the PLLB series (5.39) and PGUB series (5.43), respectively, when $\Sigma_z = 2$ and $\Sigma_y = 10$.

To demonstrate the effect of parameters Σ_z and Σ_y (and therefore $\mathbf{Z}$ and $\mathbf{Y}$), figure 5.5 shows their effect on the MSE given fixed estimators. It can be seen that Σ_z has a more prominent effect on the PLLB while Σ_y has it on the PGUB. However, it can also be observed that both parameters affect both bounds to some degree, revealing some limitations when specific bounds are desired using the proposed scheme. Figure 5.6 further captures this relation between the bounds and the parameters Σ_z and Σ_y. As the simulated system is asymptotically stable, steady-state error covariances are reached

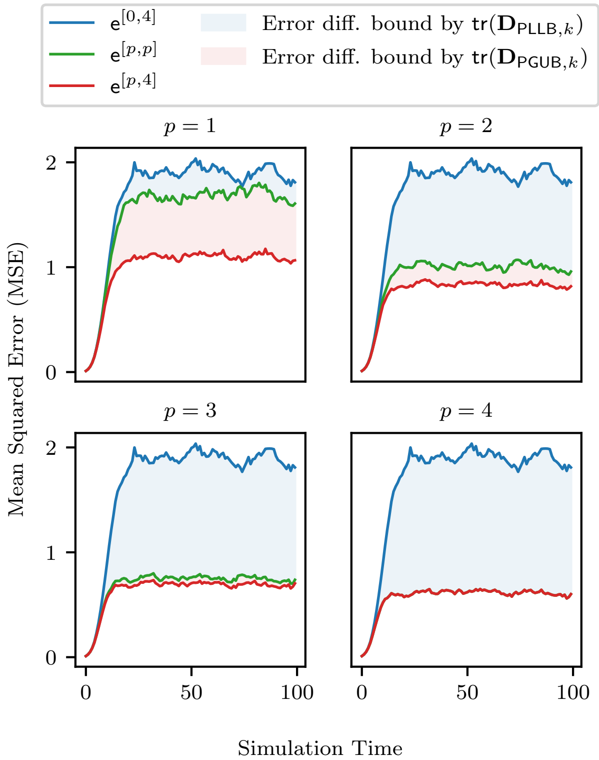


Figure 5.4.: The average errors of different estimators for 1000 simulation runs when $\Sigma_z = 2$ and $\Sigma_y = 10$.

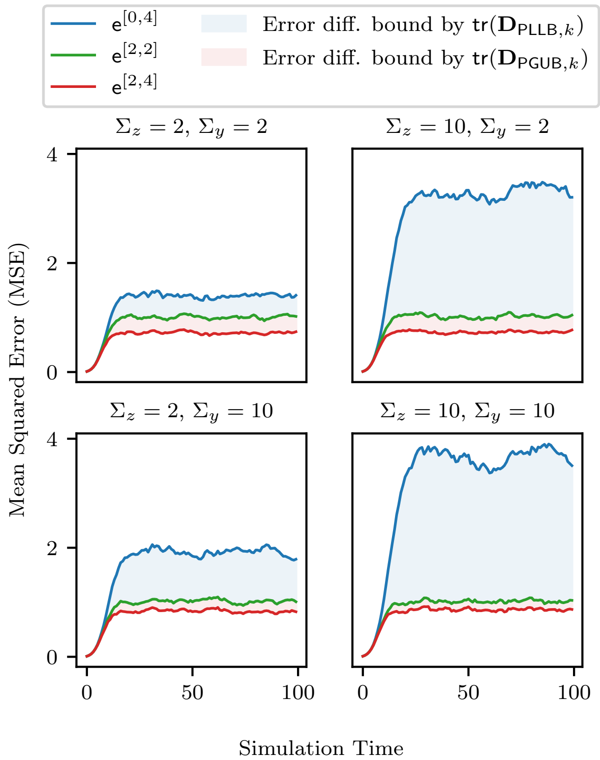


Figure 5.5.: The average errors of unprivileged and privilege-2 estimators for 1000 simulation runs when varying Σ_z and Σ_y.

as $k \to \infty$, and therefore $\mathbf{D}_{\mathsf{PLLB},k}$ and $\mathbf{D}_{\mathsf{PGUB},k}$ stabilise as well. From the plot, we can see that increasing the fully correlated noise parameter Σ_z cannot greatly reduce the PGUB (*i.e.*, bring $\mathsf{tr}(\mathbf{D}_{\mathsf{PGUB},k})$ closer to 0), likely due to the accurate estimation of this component by privileged estimators and the remaining uncorrelated component staying unchanged. Simultaneously, however, the fully correlated component can greatly increase the PLLB (*i.e.*, take $\mathsf{tr}(\mathbf{D}_{\mathsf{PLLB},k})$ further from 0) as it increases the redundancy of fusing only unprivileged measurements. The effects of increasing Σ_y are less one-sided. The PGUB is reduced due to sufficient uncorrelated noise making the fusion of unprivileged measurements hold little information even when some keys are known, but the PLLB is increased, as uncorrelated noise still affects estimators fusing only unprivileged measurements, albeit less drastically.

Figure 5.6 also shows how the bounds are affected by the privilege p they are computed for. Predictably, a higher privilege results in fewer additional unprivileged measurements to fuse, lowering the PGUB, but also producing better privileged estimates, increasing the PLLB. We can also see that when the fully correlated noise term Σ_z is small and privilege is low ($p = 1$), unprivileged estimators with access to all measurements can perform better than privileged ones accessing only privileged measurements (resulting in a negative $\mathsf{tr}(\mathbf{D}_{\mathsf{PLLB},k})$).

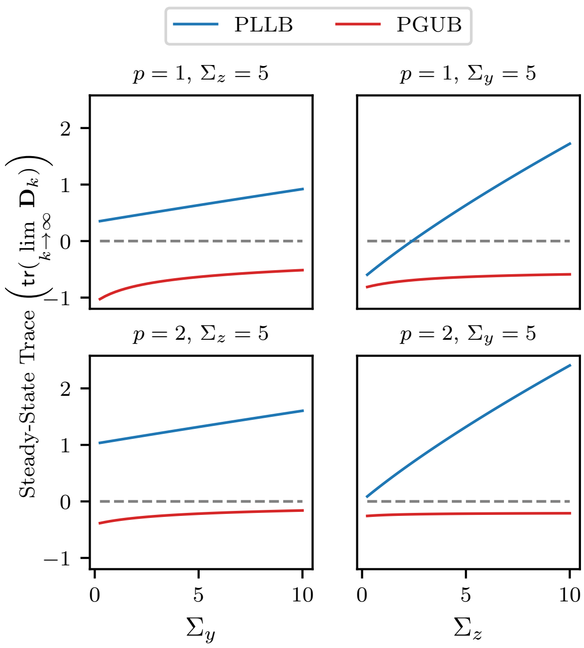


Figure 5.6.: Steady-state traces of the PLLB and PGUB for privileges $p = 1$ and $p = 2$ when Σ_z and Σ_y are varied.

5.4. Conclusions on Provable Estimation Performances

6. Conclusion

A. Linear-Combination Aggregator Obliviousness

B. Cryptographic Proof of LCAO Scheme Security

List of Figures

3.1. Approximation of ω_1 with discretization step-size $s = 0.1$. Only comparisons between line points are used. 18
3.2. Partial solutions over ω_1, ω_2, and ω_3 solution space. 20
3.3. Tracking simulation comparing SecFCI and FCI. 22
3.4. $\underline{\omega}_{SecFCI}$ and $\underline{\omega}_{FCI}$ components. 23
3.5. Average RMSE of encrypted and unencrypted FCI fusion over 1000 simulations. 28

4.1. Required linear combination aggregation steps at instance t. 30
4.2. Procedure at timestep k for the proposed privacy-preserving EIF. 42
4.3. Different simulation layouts with varying distances between navigator and sensors. 45
4.4. Average RMSE of our privacy-preserving filter and the standard EIF for different layouts. 46
4.5. Runtimes for varying key sizes and numbers of sensors. 46

5.1. Privileged estimation with bounded error covariance. 58
5.2. Privileged estimation with unbounded error covariance. 58
5.3. Estimation with multiple privilege levels. 59
5.4. The average errors of different estimators for 1000 simulation runs when $\Sigma_z = 2$ and $\Sigma_y = 10$. 68
5.5. The average errors of unprivileged and privilege-2 estimators for 1000 simulation runs when varying Σ_z and Σ_y. 69
5.6. Steady-state traces of the PLLB and PGUB for privileges $p = 1$ and $p = 2$ when Σ_z and Σ_y are varied. 70

List of Tables

3.1. Computation complexity of encryption operations. 21
3.2. Computation complexity at sensors and fusion centre. 22
3.3. Computation complexity of encryption operations. 26
3.4. Computation complexity at parties during fusion. 26

Bibliography

[1] B. D. O. Anderson and J. B. Moore, *Optimal Filtering.* Dover Publications, 1979.

[2] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches.* John Wiley & Sons, 2006.

[3] A. G. O. Mutambara, *Decentralized Estimation and Control for Multisensor Systems.* CRC press, 1998.

[4] M. Liggins, C. Y. Chong, D. Hall, and J. Llinas, *Distributed Data Fusion for Network-Centric Operations.* CRC Press, 2012.

[5] C. Y. Chong, "Forty Years of Distributed Estimation: A Review of Noteworthy Developments," in *IEEE ISIF Workshop on Sensor Data Fusion: Trends, Solutions, Applications (SDF 2017)*, 2017, pp. 1–10.

[6] A. J. Haug, *Bayesian Estimation and Tracking: A Practical Guide.* John Wiley & Sons, 2012.

[7] B. Noack, J. Sijs, M. Reinhardt, and U. D. Hanebeck, *Treatment of Dependent Information in Multisensor Kalman Filtering and Data Fusion.* CRC Press, 2017, pp. 169–192.

[8] M. Brenner, J. Wiebelitz, G. von Voigt, and M. Smith, "Secret Program Execution in the Cloud Applying Homomorphic Encryption," in *5th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, 2011, pp. 114–119.

[9] K. Ren, C. Wang, and Q. Wang, "Security Challenges for the Public Cloud," *IEEE Internet Computing*, vol. 16, no. 1, pp. 69–73, 2012.

[10] S. Gueron, "Intel Advanced Encryption Standard (AES) New Instructions Set," *Intel Corporation*, 2010.

[11] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Communications of the ACM (CACM)*, vol. 21, no. 2, pp. 120–126, 1978.

[12] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," in *Advances in Cryptology (EUROCRYPT).* Springer, 1999, pp. 223–238.

[13] E. Shi, T.-H. H. Chan, and E. Rieffel, "Privacy-Preserving Aggregation of Time-Series Data," *Annual Network & Distributed System Security Symposium (NDSS)*, p. 17, 2011.

[14] M. Joye and B. Libert, "A Scalable Scheme for Privacy-Preserving Aggregation of Time-Series Data," in *International Conference on Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science. Springer, 2013, pp. 111–125.

[15] J. Chotard, E. Dufour Sans, R. Gay, D. H. Phan, and D. Pointcheval, "Decentralized Multi-Client Functional Encryption for Inner Product," in *Advances in Cryptology (ASIACRYPT)*, ser. Lecture Notes in Computer Science. Springer, 2018, pp. 703–732.

[16] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-Indistinguishability: Differential Privacy for Location-Based Systems," in *ACM SIGSAC Conference on Computer & Communications Security*, ser. CCS '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 901–914.