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

Data Confidentiality for Distributed Sensor Fusion

Marko Ristic

March 29, 2023

Reviewers:

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

...

Contents

Acknowledgements

Acks go here. numjamhdu

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 system 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 $\underline{\hat{x}}_{k|k'} \in \mathbb{R}^d$ and $\mathbf{P}_{k|k'} \in \mathbb{R}^{d\times d}$, 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 $\underline{x}_k \in \mathbb{R}^d$ at each timestep k, following the linear system 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 $\mathbf{Q}_k \in \mathbb{R}^{d\times d}$. 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 $\underline{\hat{x}}_{0|0}$ 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

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

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

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

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 system 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 system model is now non-linear and given by

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

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

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

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

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

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

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

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

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

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

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

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 system 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 $\underline{\hat{y}}_{k|k'}$ and information matrix $\mathbf{Y}_{k|k'}$ rather than the estimate and its error covariance, $\underline{\hat{x}}_{k|k'}$ and $\mathbf{P}_{k|k'}$, 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

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

and

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

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

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

and

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

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

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

and

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

Since all information related to measurements and their sensors are captured in $\underline{i}_k$ and $\mathbf{I}_k$, namely the measured value $\underline{z}_k$, 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 system 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 system model (2.8) and multiple sensors i, $1 \leq i \leq n$, making independent non-linear measurements that follow models

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

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

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

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 system model

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

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

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

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

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

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, IF, EKF or EIF instances, that may contain cross-correlations due to shared system model assumptions during estimation [ci and dependence ref]. Covariance intersection (CI), is a time-independent fusion algorithm that fuses estimates or measurements and their 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. CI is particularly well suited to fusing estimates $\hat{\underline{x}}_i$ and covariances $\mathbf{P}_i$ of sensor i in the information form, as given in (2.16) and (2.17), such that the fusion of n information vectors $\mathbf{P}_i^{-1}\hat{\underline{x}}_i$, $1 \leq i \leq n$, and information matrices $\mathbf{P}_i^{-1}$, $1 \leq i \leq n$, produces a fused information vector $\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}}$ and matrix $\mathbf{P}_{\mathsf{fus}}^{-1}$. It is given by

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

and

$$\mathbf{P}_{\mathsf{fus}}^{-1} = \sum_{i=1}^{n} \omega_i \mathbf{P}_i^{-1} \,, \tag{2.39}$$

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 $\mathbf{P}_{\mathsf{fus}}$. 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 (2.41), 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.* (Generate, Encrypt, 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 (Generate, Encrypt, 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 (Setup, Encrypt, 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 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 (Setup, Encrypt, 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 $\mathsf{pk} = N$ and the secret key by $\mathsf{sk} = (p, q)$.

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

$$c = \mathcal{E}_{\mathsf{pk}}(a) = (N+1)^a \rho^N \pmod{N^2} \tag{2.45}$$

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 the ciphertext c is computed by

$$a = \mathcal{D}_{\mathsf{pk},\mathsf{sk}}(c) = \frac{L(c^\lambda \pmod{N^2})}{L((N+1)^\lambda \pmod{N^2})} \pmod{N} \tag{2.46}$$

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

In addition to encryption and decryption, the Paillier scheme provides the following homomorphic properties, $\forall a_1, a_2 \in \mathbb{Z}_N$,

$$\mathcal{D}_{\mathsf{pk},\mathsf{sk}}\left(\mathcal{E}_{\mathsf{pk}}(a_1)\mathcal{E}_{\mathsf{pk}}(a_2) \pmod{N^2}\right) = a_1 + a_2 \pmod{N}, \tag{2.47}$$

$$\mathcal{D}_{\mathsf{pk},\mathsf{sk}}\left(\mathcal{E}_{\mathsf{pk}}(a_1)(N+1)^{a_2} \pmod{N^2}\right) = a_1 + a_2 \pmod{N}, \tag{2.48}$$

$$\mathcal{D}_{\mathsf{pk},\mathsf{sk}}\left(\mathcal{E}_{\mathsf{pk}}(a_1)^{a_2} \pmod{N^2}\right) = a_1 a_2 \pmod{N}. \tag{2.49}$$

To simplify this notation, the shorthand operators $\oplus$ and $\otimes$ are used to denote homomorphic addition and multiplication, respectively, as

$$\mathcal{E}_{\mathsf{pk}}(a_1) \oplus \mathcal{E}_{\mathsf{pk}}(a_2) \equiv \mathcal{E}_{\mathsf{pk}}(a_1)\mathcal{E}_{\mathsf{pk}}(a_2) \pmod{N^2} \tag{2.50}$$

and

$$a_2 \otimes \mathcal{E}_{\mathsf{pk}}(a_1) \equiv \mathcal{E}_{\mathsf{pk}}(a_1)^{a_2} \pmod{N^2}. \tag{2.51}$$

Similarly, encryption $\mathcal{E}_{\mathsf{pk}}(\cdot)$, decryption $\mathcal{D}_{\mathsf{pk},\mathsf{sk}}(\cdot)$ and the above operators $\oplus$ and $\otimes$ will denote elementwise operations when inputs are multi-dimensional.

2.2.3. Joye-Libert Aggregation Scheme

The Joye-Libert privacy-preserving aggregation scheme [joyeScalableSchemePrivacyPreserving2013] is a scheme defined for $n+1$ parties, where n participants have their data aggregated by an aggregator that performs the homomorphic summation. It is defined on time-series data, indexed by timestep k, 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 secret keys for individual participants and the aggregating

party, requiring an additional trusted party to perform an initial key generation and distribution step.

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 scheme's public parameters are set to $\mathsf{pub} = (N, H)$. n secret keys are generated by choosing sk_i, $1 \leq i \leq n$, uniformly from $\mathbb{Z}_{N^2}$ and distributing them to n participants (whose data is to be aggregated). The last secret key is set as

$$\mathsf{sk}_0 = -\sum_{i=1}^{n} \mathsf{sk}_i \,, \tag{2.52}$$

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

$$c_i^{(k)} = \mathcal{E}_{\mathsf{sk}_i}\left(a_i^{(k)}\right) = (N+1)^{a_i^{(k)}} H(k)^{\mathsf{sk}_i} \pmod{N^2}. \tag{2.53}$$

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

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

$$c^{(k)} = \prod_{i=1}^{n} c_i^{(k)} \pmod{N^2} \tag{2.54}$$

and

$$\sum_{i=1}^{n} a_i^{(k)} = \mathcal{D}_{\mathsf{sk}_0}\left(c^{(k)}\right) = \frac{H(k)^{\mathsf{sk}_0} c^{(k)} - 1}{N} \pmod{N}. \tag{2.55}$$

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

$$\begin{aligned}
& H(k)^{\mathsf{sk}_0} \prod_{i=1}^{n} c_i^{(k)} \pmod{N^2} \\
\equiv & H(k)^{\mathsf{sk}_0} \prod_{i=1}^{n} (N+1)^{a_i^{(k)}} H(k)^{\mathsf{sk}_i} \pmod{N^2} \\
\equiv & H(k)^{\sum_{j=0}^{n} \mathsf{sk}_j} \prod_{i=1}^{n} (N+1)^{a_i^{(k)}} \pmod{N^2} \\
\equiv & (N+1)^{\sum_{i=1}^{n} a_i^{(k)}} \pmod{N^2},
\end{aligned}$$

removing all noise terms.

2.2.4. Lewi Order-Revealing Encryption Scheme

The Lewi order-revealing encryption (ORE) 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 its notation will be introduced here. The two encryption equations provided can encrypt plaintexts $a_1, a_2 \in \mathbb{Z}$ with a secret key $\mathsf{sk_o}$ and functions

$$\mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(a_1) \text{ and } \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(a_2) \tag{2.56}$$

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

$$\mathcal{C}\left(\mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(a_1), \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(a_2)\right) = \mathsf{cmp}(a_1, a_2)\,, \tag{2.57}$$

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 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 (2.54). 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] 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

$$\mathbb{Q}_{M,\phi} = \left\{ \chi \,\middle|\, \phi\chi \in \mathbb{N} \wedge -\left\lfloor \frac{M}{2} \right\rfloor \leq \phi\chi < \left\lfloor \frac{M}{2} \right\rfloor \right\}, \tag{2.58}$$

and can be used to quantise any real number $a \in \mathbb{R}$ by taking the nearest rational $\chi \in \mathbb{Q}_{M,\phi}$, that is, $\arg\min_{\chi \in \mathbb{Q}_{M,\phi}} |a - \chi|$. 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 δ; the number of additional precision factors present in encodings.

The function for *combined* quantisation and encoding, $\mathsf{E}_{M,\phi,\delta}(a)$, of a given number $a \in \mathbb{R}$ and with output integer range $\mathbb{Z}_M$, precision ϕ and scaling for δ prior encoded multiplications, is given by

$$\mathsf{E}_{M,\phi,\delta}(a) = \left\lfloor \phi^{\delta+1} a \right\rceil \pmod{M}. \tag{2.59}$$

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

$$\mathsf{E}^{-1}_{M,\phi,\delta}(u) = \begin{cases} \dfrac{u \pmod{M}}{\phi^{\delta+1}}, & u \pmod{M} \leq \left\lfloor \dfrac{M}{2} \right\rfloor \\ -\dfrac{M - u \pmod{M}}{\phi^{\delta+1}}, & \text{otherwise} \end{cases}. \tag{2.60}$$

Encoding with (2.59) additionally provides two useful properties when used with homomorphic operations,

$$\mathsf{E}_{M,\phi,\delta}(a_1) + \mathsf{E}_{M,\phi,\delta}(a_2) \pmod{M} \approx \mathsf{E}_{M,\phi,\delta}(a_1 + a_2) \tag{2.61}$$

and

$$\mathsf{E}_{M,\phi,\delta}(a_1)\mathsf{E}_{M,\phi,\delta}(a_2) \pmod{M} \approx \mathsf{E}_{M,\phi,\delta+1}(a_1 a_2), \tag{2.62}$$

where deviation from equality stems from quantisation and its associated error

$$\left| \mathsf{E}^{-1}_{M,\phi,\delta}\left(\mathsf{E}_{M,\phi,\delta}(a)\right) - a \right| \leq \frac{1}{\phi}. \tag{2.63}$$

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, δ_{max}, and the largest value to be encoded, a_{max}, are known, ϕ can be chosen to avoid overflow by satisfying

$$\left| \phi^{\delta_{\mathsf{max}}+1} a_{\mathsf{max}} \right| < \left\lfloor \frac{M}{2} \right\rfloor. \tag{2.64}$$

In practice, we set $M = N$ to use the Paillier or Joye-Libert schemes, such that the properties (2.61) and (2.62) can be used with the homomorphic operations of the schemes.

Equation (2.64) 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. Lastly, as with the simplified notation of Paillier functions, encoding $\mathsf{E}_{M,\phi,\delta}(\cdot)$ and decoding $\mathsf{E}^{-1}_{M,\phi,\delta}(\cdot)$ will denote elementwise operations when inputs are multi-dimensional.

3. Estimate Fusion on an Untrusted Cloud

3.1. Problem Formulation

Motivated by the key step in multi-sensor fusion, we are interested in transmitting local sensor state estimate and covariance information over a network to be fused by a fusion cloud. In particular, we consider centralised FCI fusion, as introduced in section 2.1.6, over a public network and at an untrusted fusion cloud. The aim is for fusion to be computed by the cloud on encrypted sensor data to preserve individual and fused estimate confidentiality, that is, no estimate information should be made available to network eavesdroppers, sensors that did not produce it or the fusion cloud itself. A trusted querying third party, holding appropriate secret keys, is presumed to exist which can request and process the fused estimate information from the cloud.

The concrete estimation problem is captured by a time-independent process defined by its state $\underline{x} \in \mathbb{R}^d$ and is estimated by sensors i, $1 \leq i \leq n$, each producing a state estimate and estimate error covariance,

$$\hat{\underline{x}}_i \in \mathbb{R}^d \text{ and } \mathbf{P}_i \in \mathbb{R}^{d \times d}\,, \tag{3.1}$$

respectively. As we are interested in computing the FCI algorithm, we consider sensors that produce estimates in the information form, namely, the information vector and information matrix,

$$\mathbf{P}_i^{-1}\hat{\underline{x}}_i \text{ and } \mathbf{P}_i^{-1}\,, \tag{3.2}$$

instead. Information (3.2) is to be sent to the fusion cloud where FCI is computed and a fused information vector and information matrix,

$$\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}} \in \mathbb{R}^d \text{ and } \mathbf{P}_{\mathsf{fus}}^{-1} \in \mathbb{R}^{d \times d}\,, \tag{3.3}$$

are produced. A trusted querying party can then request (3.3) from the cloud when desired.

The cryptographic aims of this problem are captured by the actions of the involved parties and the accepted leakage of confidential information.

Honest-but-curious parties We assume that all sensors and the fusion cloud follow fusion protocols correctly and that no injection or modification to transmitted data is performed by network eavesdroppers, but all parties may use any learned information for external malicious gain. Collusion between any malicious parties is considered possible.

Encrypted estimates meeting IND-CPA Permitted leakage is such that all estimate in-

formation available to a colluding subset of malicious parties, excluding any locally produced estimates (in the case of malicious sensors), is encrypted with a scheme meeting the IND-CPA cryptographic notion, introduced in section 2.2.1.

The relatively strict estimation and security aims above can be difficult to achieve in general and relaxations of some requirements may be necessary to achieve others. This will be seen in the two methods presented later in this chapter. Participants, communications between them and whether they are trusted, concerning the ideal aims above, are summarised graphically in figure 3.1.

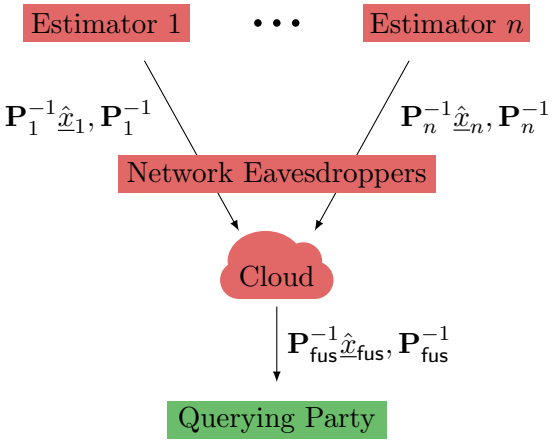


Figure 3.1.: Trusted (green) and untrusted (red) participants, and the communications between them in the cloud fusion problem.

3.2. Confidential Cloud Fusion Leaking Fusion Weights

In this section, we present a method for computing the fusion result (3.3) at the fusion cloud by leaking the FCI fusion weights (2.44) to malicious adversaries. As stated in the problem formulation, solving the fusion problem, in general, is difficult, and we make some relaxations to the cryptographic aims for this solution.

Trusted sensors In this method, we assume that sensors are trusted. That is, only the fusion cloud and eavesdroppers are considered honest-but-curious adversaries, and no sensor estimate information should be made available to them.

Leakage of fusion weights While all estimate information available to colluding malicious parties should be encrypted with a scheme meeting the IND-CPA notion, we make an exception for the FCI fusion weights (2.44), which may be leaked to both the fusion cloud and eavesdroppers.

Here, we also note that the weakening of cryptographic guarantees caused by the leakage of fusion weights also has an upside that benefits fusion performance. The leakage of

weights allows the cloud to prioritise some sensor data over others. For example, in bandwidth-limited networks, communication with sensors of lower weight, and therefore high uncertainty, may be dropped in favour of those that lead to better fusion results.

Lastly, we assume that the trusted sensors are computationally capable of locally running both the Paillier and Lewi encryption schemes, introduced in sections 2.2.2 and 2.2.4, and require a one-time network key distribution step before fusion of sensor data can take place at the cloud. This key distribution step consists of a trusted party generating a Paillier scheme public key pair pk and sk and a shared Lewi scheme symmetric key $\mathsf{sk_o}$. The public key pk is made available to the cloud and sensors, the secret key sk to the querying party and the shared ORE key $\mathsf{sk_o}$ to the sensors. The sharing of these keys can be performed by any public-key encryption scheme such as RSA [rivest-MethodObtainingDigital1978].

3.2.1. Two-sensor Case

The confidential fusion algorithm will first be introduced in the special case of two sensors, before an extension to the n-sensor case. Recalling FCI fusion (2.38), (2.39) and (2.40), we can write the fusion of two information vectors and matrices as

$$\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}} = \omega_1\mathbf{P}_1^{-1}\hat{\underline{x}}_1 + (1-\omega_1)\mathbf{P}_2^{-1}\hat{\underline{x}}_2 \tag{3.4}$$

and

$$\mathbf{P}_{\mathsf{fus}}^{-1} = \omega_1\mathbf{P}_1^{-1} + (1-\omega_1)\mathbf{P}_2^{-1}\,, \tag{3.5}$$

with $0 \leq \omega_1 \leq 1$, and note the suitability of addition and scalar multiplication to the Paillier scheme when the weight ω_1 is known. To compute the fused information vector $\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}}$ and information matrix $\mathbf{P}_{\mathsf{fus}}^{-1}$ homomorphically, local estimate information must first be encoded as integers before encryption at the sensors. Using the Q number format in section 2.2.5, we let $M = N$, where N is the Paillier modulus, choose an appropriate precision ϕ and denote encoding with δ previous multiplications as $\mathsf{E}_\delta(\cdot)$. Encoding, encryption and fusion with Paillier homomorphic properties (2.47) and (2.49) is then given by

$$\mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_1\left(\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}}\right)\right) \approx \left(\mathsf{E}_0(\omega_1) \otimes \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\mathbf{P}_1^{-1}\hat{\underline{x}}_1\right)\right)\right) \oplus \left(\mathsf{E}_0(1-\omega_1) \otimes \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\mathbf{P}_2^{-1}\hat{\underline{x}}_2\right)\right)\right) \tag{3.6}$$

and

$$\mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_1\left(\mathbf{P}_{\mathsf{fus}}^{-1}\right)\right) \approx \left(\mathsf{E}_0(\omega_1) \otimes \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\mathbf{P}_1^{-1}\right)\right)\right) \oplus \left(\mathsf{E}_0(1-\omega_1) \otimes \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\mathbf{P}_2^{-1}\right)\right)\right)\,, \tag{3.7}$$

with encoding approximation errors dependent on precision parameter ϕ. The trusted querying party can now request encryptions (3.6) and (3.7) from the cloud and decrypt them with secret key sk.

All that remains for computing (3.6) and (3.7) in the two-sensor case is obtaining the parameter ω_1 at the cloud. Since ω_1 depends on the estimate errors of both sensors it cannot be computed locally and requires appropriate leakage to the cloud. This is

achieved by using the Lewi ORE scheme and encrypting discretised sequences whose intersection can be used to compute the two-sensor FCI fusion weight constraint (2.42), namely

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

To evaluate (3.8) with comparisons at the cloud, a public stepsize $g \leq 1$ is chosen, such that $1/g \in \mathbb{N}$, and both sensors discretise the weight $0 \leq \omega_1 \leq 1$ and compute resulting sequences of either side of the equality in (3.8). This results in the sequence

$$\langle 0, g\operatorname{tr}(\mathbf{P}_1), 2g\operatorname{tr}(\mathbf{P}_1), \ldots, \operatorname{tr}(\mathbf{P}_1)\rangle \tag{3.9}$$

at sensor 1 and

$$\langle \operatorname{tr}(\mathbf{P}_2), (1-g)\operatorname{tr}(\mathbf{P}_2), (1-2g)\operatorname{tr}(\mathbf{P}_2), \ldots, 0\rangle \tag{3.10}$$

at sensor 2. Comparison of same-index values in the sequences (3.9) and (3.10) leads to the bounds $\iota g < \omega_1 < (\iota+1)g$, for some index ι, that can be used to approximate the true solution as $\hat{\omega}_1 = \iota g + g/2 \approx \omega_1$, or in the case of an equality, $\hat{\omega}_1 = \iota g = \omega_1$. To obtain this approximation without additional leakage, elements in (3.9) and (3.10) are encrypted with the ORE key $\mathsf{sk_o}$. As no homomorphic operations are performed with the scheme an arbitrary precision integer encoding can be used and is neglected from the notation below. The sequence produced by sensor 1 is therefore given by

$$\left\langle \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(0), \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(g\operatorname{tr}(\mathbf{P}_1)), \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(2g\operatorname{tr}(\mathbf{P}_1)), \ldots, \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(\operatorname{tr}(\mathbf{P}_1))\right\rangle \tag{3.11}$$

and that by sensor 2 by

$$\left\langle \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(\operatorname{tr}(\mathbf{P}_2)), \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}((1-g)\operatorname{tr}(\mathbf{P}_2)), \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}((1-2g)\operatorname{tr}(\mathbf{P}_2)), \ldots, \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(0)\right\rangle\,, \tag{3.12}$$

where we note the difference between *left* and *right* encryptions for each sensor, allowing comparisons between sequences. Efficiently findable using a binary search, the approximate solution when comparing elements of the two sequences can be seen graphically in figure 3.2, where the approximation is taken as halfway between consecutive comparisons that change sign.

In this way, computing fusion on the cloud homomorphically can be performed when having access to only local encryptions of information vectors and matrices, (3.7) and (3.6), in addition to the sequences of order-revealing encryptions (3.11) and (3.12) that leak an approximation $\hat{\omega}_1 \approx \omega_1$.

3.2.2. Multi-sensor Case

To compute the fusion of n sensor estimates in the information form we want the solutions to

$$\mathbf{P}^{-1}_{\mathsf{fus}}\hat{\underline{x}}_{\mathsf{fus}} = \sum_{i=1}^{n} \omega_i \mathbf{P}_i^{-1}\hat{\underline{x}}_i \tag{3.13}$$

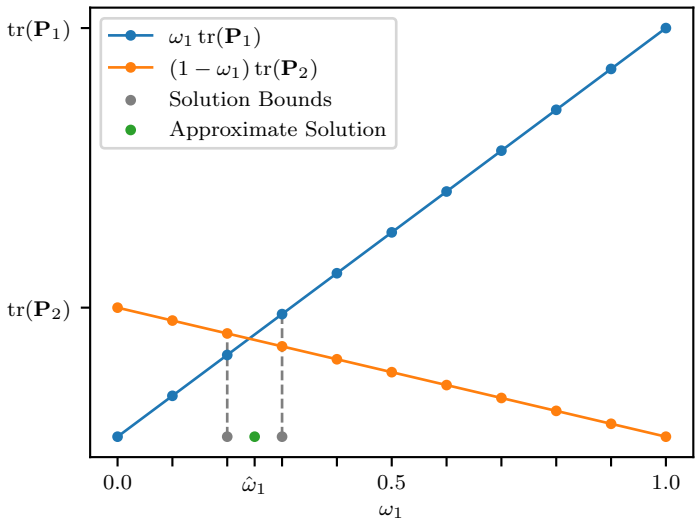


Figure 3.2.: Approximation of ω_1 with stepsize $g = 0.1$. Comparisons are only possible when ω_1 is a multiple of g (points on the graphs).

and

$$\mathbf{P}_{\mathsf{fus}}^{-1} = \sum_{i=1}^{n} \omega_i \mathbf{P}_i^{-1} \,. \tag{3.14}$$

When the fusion weights ω_i, $1 \leq i \leq n$, $\sum_{i=1}^{n} \omega_i = 1$, are known, we can again use the appropriate integer encoding and Paillier homomorphic properties to compute the fusion homomorphically as

$$\mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_1\left(\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}}\right)\right) \approx \oplus_{i=1}^{n}\left(\mathsf{E}_0(\omega_i) \otimes \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\mathbf{P}_i^{-1}\hat{\underline{x}}_i\right)\right)\right) \tag{3.15}$$

and

$$\mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_1\left(\mathbf{P}_{\mathsf{fus}}^{-1}\right)\right) \approx \oplus_{i=1}^{n}\left(\mathsf{E}_0(\omega_i) \otimes \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\mathbf{P}_i^{-1}\right)\right)\right) \,. \tag{3.16}$$

What remains is to compute the weights ω_i at the cloud such that the n-sensor FCI conditions (2.42) are met. Similar to the two-sensor case, we can use sequences of order-revealing encryptions to leak an approximation to this result.

Each condition (2.42) is considered as a partial problem, that is,

$$\omega_i \operatorname{tr}(\mathbf{P}_i) = \omega_{i+1} \operatorname{tr}(\mathbf{P}_{i+1}) \,, \tag{3.17}$$

with $1 \leq i < n$ and $\sum_{i=1}^{n} \omega_i = 1$, and their solution spaces, linear subspaces $\mathcal{S}_i$ over possible values of $\underline{\omega} = [\,\omega_1 \;\cdots\; \omega_n\,]^\top$, desired. The intersection of these subspaces naturally results in the final solution to $\underline{\omega}$ in (2.43). Each solution space $\mathcal{S}_i$ can be defined by $n-1$ linearly independent solution points $\underline{\omega}_i^{(\zeta)}$, $1 \leq \zeta \leq n-1$. $n-2$ of these solutions can

be trivially obtained as the points where a single $\omega_j = 1$, $j \neq i \neq i+1$ while the final point, $\underline{\omega}_i^{(n-1)}$, can be obtained as the solution to

$$\omega_i \operatorname{tr}(\mathbf{P}_i) = (1-\omega_i)\operatorname{tr}(\mathbf{P}_{i+1}), \tag{3.18}$$

$\omega_j = 0$, $j \neq i \neq i+1$, noting equivalence to (3.17) and $\omega_{i+1} = 1-\omega_i$. The solutions $\underline{\omega}_i^{(\zeta)}$ and the $n-2$ dimensional subspace they define can be rewritten in parametric form as

$$\mathcal{S}_i(\underline{\gamma}) = \underline{\omega}_i^{(1)} + \begin{bmatrix} \bar{\underline{\omega}}_i^{(2)} & \cdots & \bar{\underline{\omega}}_i^{(n-1)} \end{bmatrix} \underline{\gamma}, \tag{3.19}$$

where $\bar{\underline{\omega}}_i^{(\zeta)} = \underline{\omega}_i^{(\zeta)} - \underline{\omega}_i^{(1)}$ denote the direction vectors. The intersection of these $n-1$ subspaces gives the solution to $\underline{\omega}$ in (2.43). This solution, as well as the parameters for each subspace that provide it, $\underline{\gamma}_i$, can be obtained as a solution to the linear system

$$\begin{bmatrix} -\mathbf{I} & \bar{\underline{\omega}}_1^{(2)} & \cdots & \bar{\underline{\omega}}_1^{(n-1)} & 0 & \cdots & 0 \\ \vdots & 0 & & \ddots & & \ddots & \vdots \\ \vdots & \vdots & \ddots & & \ddots & & 0 \\ -\mathbf{I} & 0 & \cdots & 0 & \bar{\underline{\omega}}_{n-1}^{(2)} & \cdots & \bar{\underline{\omega}}_{n-1}^{(n-1)} \end{bmatrix} \begin{bmatrix} \underline{\omega} \\ \underline{\gamma}_1 \\ \vdots \\ \underline{\gamma}_{n-1} \end{bmatrix} = \begin{bmatrix} -\underline{\omega}_1^{(1)} \\ \vdots \\ -\underline{\omega}_{n-1}^{(1)} \end{bmatrix}. \tag{3.20}$$

Therefore, to compute (3.20) at the cloud leaking only comparisons required to compute the fusion weights, we approximate the solutions to (3.18) with ORE sequences similar to the two-sensor case. Each sensor i uses $\mathsf{sk_o}$ to encrypt the discretisation

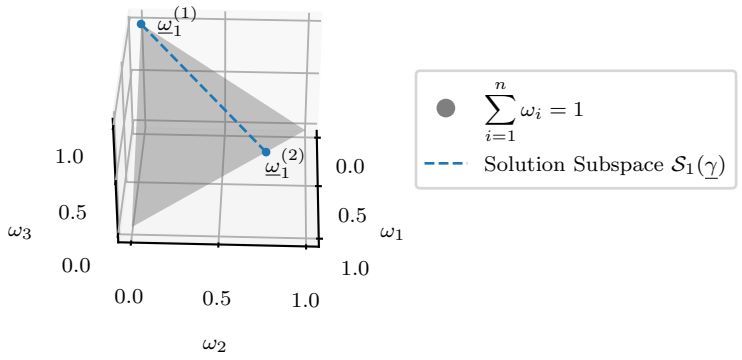
$$\begin{aligned} &\left\langle \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(0), \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(g\operatorname{tr}(\mathbf{P}_i)), \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(2g\operatorname{tr}(\mathbf{P}_i)), \ldots, \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(\operatorname{tr}(\mathbf{P}_i)) \right\rangle \text{, if } i \text{ is odd, or} \\ &\left\langle \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(0), \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(g\operatorname{tr}(\mathbf{P}_i)), \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(2g\operatorname{tr}(\mathbf{P}_i)), \ldots, \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(\operatorname{tr}(\mathbf{P}_i)) \right\rangle \text{, if } i \text{ is even,} \end{aligned} \tag{3.21}$$

allowing for comparison between sequences from consecutive sensors i and $i+1$. To approximate the solution to each (3.18), the sequence from sensor $i+1$ is reversed,

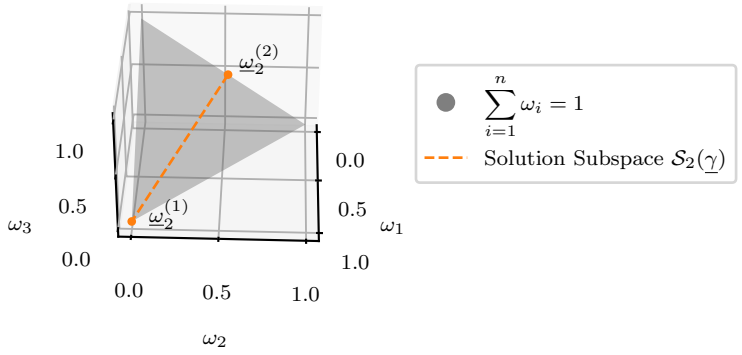
$$\begin{aligned} &\left\langle \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(\operatorname{tr}(\mathbf{P}_i)), \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}((1-g)\operatorname{tr}(\mathbf{P}_i)), \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}((1-2g)\operatorname{tr}(\mathbf{P}_i)), \ldots, \mathcal{E}^{\mathsf{L}}_{\mathsf{sk_o}}(0) \right\rangle \text{, if } i \text{ is odd, or} \\ &\left\langle \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(\operatorname{tr}(\mathbf{P}_i)), \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}((1-g)\operatorname{tr}(\mathbf{P}_i)), \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}((1-2g)\operatorname{tr}(\mathbf{P}_i)), \ldots, \mathcal{E}^{\mathsf{R}}_{\mathsf{sk_o}}(0) \right\rangle \text{, if } i \text{ is even,} \end{aligned} \tag{3.22}$$

resulting in sequences of the same form as (3.11) and (3.12). The value $\hat{\omega}_i = \iota g + g/2 \approx \omega_i$, or in the case of an equality $\hat{\omega}_i = \iota g = \omega_i$, for some index ι, is used to approximate the solution $\hat{\underline{\omega}}_i^{(n-1)} \approx \underline{\omega}_i^{(n-1)}$. A visualisation of solving (3.20) with partial solutions (3.18) in the three-sensor case is graphically shown in figure 3.3.

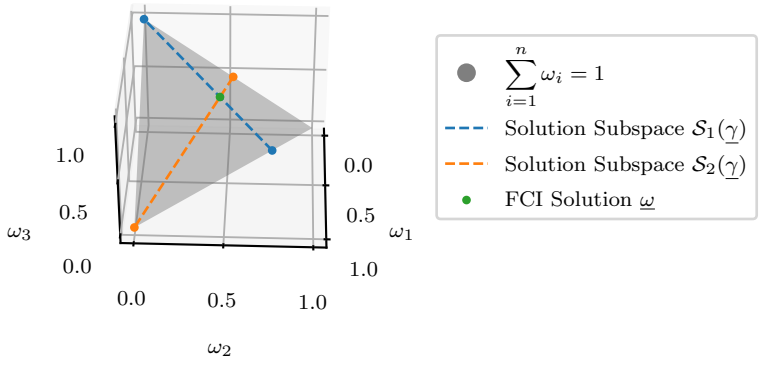
A resulting estimate $\hat{\underline{\omega}} \approx \underline{\omega}$ can then be used with (3.15) and (3.16) to compute the fusion of n information vectors and matrices at the cloud.



(a) Approximated solution space (3.19) when $i = 1$.



(b) Approximated solution space (3.19) when $i = 2$.



(c) Intersection of partial solutions spaces gives fusion weights $\underline{\omega}$ in (2.43).

Figure 3.3.: Solving fusion weights $\underline{\omega}$ with approximations to (3.18).

3.2.3. Computational Complexity

The method introduced allows the computation of FCI by using the Paillier and Lewi encryption schemes. Naturally, relying on encryption and homomorphic operations increases the complexity of computing this fusion at the sensor, cloud and querying party. Here, we look at the computational complexity of the method in terms of the required operations at each party. We assume that both the Lewi and Paillier schemes use the same key size, typically measured in bits, $\log N$, where N is the Paillier modulus defined in section 2.2.2. In addition, we note the distinction between floating-point or small integer operations, treated as having runtime $O(1)$, and large integer operations with runtime dependent on bit length. While hardware architecture exists for faster encryption operations [gueronIntelAdvancedEncryption2010], we consider software implementations and treat large integer operations in terms of bit operations explicitly.

Individual encryption operation complexities with the assumptions made above have been summarised in table 3.1. Applying operation complexities to the fusion algorithm

Table 3.1.: Computation complexity of involved encryption operations.

Operation	**Complexity**
Paillier Encryption	$O(\log^3 N)$
Paillier Decryption	$O(\log^3 N)$
Paillier Addition	$O(\log^2 N)$
Paillier Scalar Multiplication	$O(\log^3 N)$
Lewi *Left* Encryption	$O(\log^2 N)$
Lewi *Right* Encryption	$O(\log^2 N)$
Lewi Comparison	$O(\log^2 N)$

we get the computational complexity for the sensors, cloud and querying party. These can be seen in table 3.2, where unencrypted FCI algorithm complexities are also shown for reference. These complexities show the additional computational cost required for

Table 3.2.: Computation complexity for each party.

	FCI	**Confidential Fusion Leaking Weights**
Sensor	$O(1)$	$O\left(d^2 \log^3 N + \frac{1}{g} \log^2 N\right)$
Cloud	$O\left(nd^2 + n^3\right)$	$O\left(nd^2 \log^3 N + n \log \frac{1}{g} + n^3\right)$
Querying Party	$O(1)$	$O\left(d^2 \log^3 N\right)$

providing the security benefits of the presented scheme and must naturally be reflected in chosen hardware when developing a system where the benefits are desired.

3.2.4. Security Analysis

Recalling the cryptographic aims of the estimate fusion problem introduced in the problem formulation and weakened for the presented method, we desire estimate information

to be encrypted with a notion meeting IND-CPA at the fusion cloud and eavesdroppers while allowing the leakage of fusion weights, and naturally any functions derivable only from this information.

Since Paillier encryptions (3.15) and (3.16) meet the IND-CPA notion, this information meets the desired aims. The remaining available information to the cloud and eavesdroppers are the sequences encrypted by the Lewi ORE scheme. This scheme meets the simulation-based security discussed in section 2.2.4 and is shown in section 3.2.2 to correspond to the leakage of approximations to the weights ω_i, in addition to some leakage beyond the used ciphertext order comparisons due to the relaxation of the IND-OCPA notation (which can be difficult to quantify in context). Lastly, we also acknowledge an implicit leakage of estimate dimension d associated with the use of elementwise encryption. Although methods for homomorphic encryption of vectors exist [alexandruPrivateWeightedSum2020], we leave this as future work on this topic and note that the dimension d may leak information about the data fusion use case but that estimates themselves remain encrypted.

The leakages present in the introduced method, namely the leakage of fusion weights and estimate dimensions, may lead to inferences about sensor hardware and must naturally be considered when planning the implementation of this method in a real-world system.

3.2.5. Simulation

Along with the discussions above, we have implemented a simulation of the method to demonstrate its fusion accuracy when compared to the FCI algorithm on a trusted cloud. A constant-velocity linear system,

$$\underline{x}_{k+1} = \begin{bmatrix} 1 & 0 & 0.5 & 0 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \underline{x}_k + \underline{w}_k \,, \tag{3.23}$$

with noise term

$$\underline{w}_k \sim \mathcal{N}\left(\underline{0},\ \frac{1}{10^3} \cdot \begin{bmatrix} 0.42 & 0 & 1.25 & 0 \\ 0 & 0.42 & 0 & 1.25 \\ 1.25 & 0 & 5.0 & 0 \\ 0 & 1.25 & 0 & 5.0 \end{bmatrix} \right) \,, \tag{3.24}$$

was simulated and measured by four independent position sensors $1 \leq i \leq 4$, that produced measurements

$$\underline{z}_{k,i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \underline{x}_k + \underline{v}_{k,i} \,, \qquad \underline{v}_{k,i} \sim \mathcal{N}\left(\underline{0}, \mathbf{R}_i\right) \,, \tag{3.25}$$

with noise term covariances sampled independently, resulting in

$$\begin{aligned}\mathbf{R}_1 &= \begin{bmatrix} 4.77 & -0.15 \\ -0.15 & 4.94 \end{bmatrix}, \\ \mathbf{R}_2 &= \begin{bmatrix} 2.99 & -0.55 \\ -0.55 & 4.44 \end{bmatrix}, \\ \mathbf{R}_3 &= \begin{bmatrix} 2.06 & 0.68 \\ 0.68 & 1.96 \end{bmatrix} \text{ and} \\ \mathbf{R}_4 &= \begin{bmatrix} 1.17 & 0.80 \\ 0.80 & 0.64 \end{bmatrix}.\end{aligned} \tag{3.26}$$

Each sensor ran a local linear Information Filter (IF) from section 2.1.4, initialised with the true state of the system, before processing and sending its estimate information, $\mathbf{P}_{k,i}^{-1}\hat{\underline{x}}_{k,i}$ and $\mathbf{P}_{k,i}^{-1}$, to the cloud for fusion. At each timestep, the time-independent fusion of information vectors and matrices was computed homomorphically at the cloud and decrypted by the querying party. The simulation was written in the Python and C programming languages, using the PHE Paillier encryption scheme library [Python-Paillier2013], and using a key size of 512 bits for encryption schemes. Figure 3.4 shows the average estimation error of 1000 simulations when using our algorithm with varying stepsizes g alongside the standard FCI algorithm. In all cases, resulting plaintext infor-

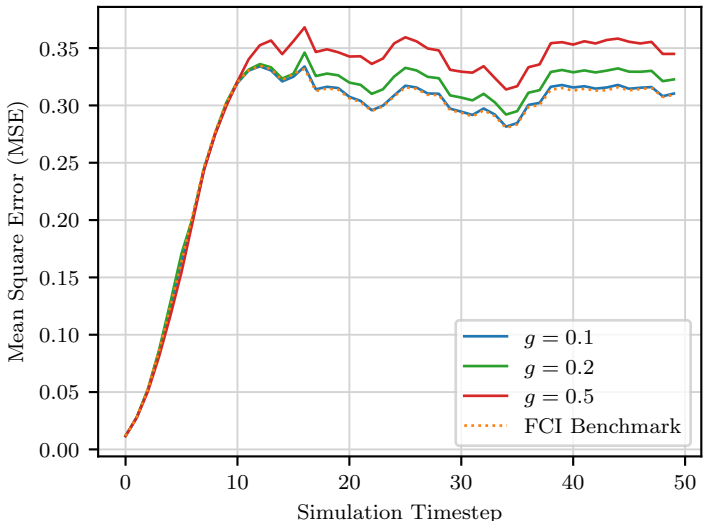


Figure 3.4.: Average MSE with varying stepsize g over 1000 simulation runs.

mation vectors and matrices were converted to estimates and estimate error covariances before comparison with the true simulated state. From the figure, it can be seen that the estimation error of our method is similar to the normal FCI method when stepsize g is small, $g = 0.1$, but grows as expected when g, and thus the possible error in weights

ω_i is increased.

Since the system described by (3.23), (3.25) and estimated by the IF reaches an estimation steady-state, fusion weights ω_i do as well. Figure 3.5 shows the steady state error of estimated weights when compared to the true FCI weights. Here, the maximum

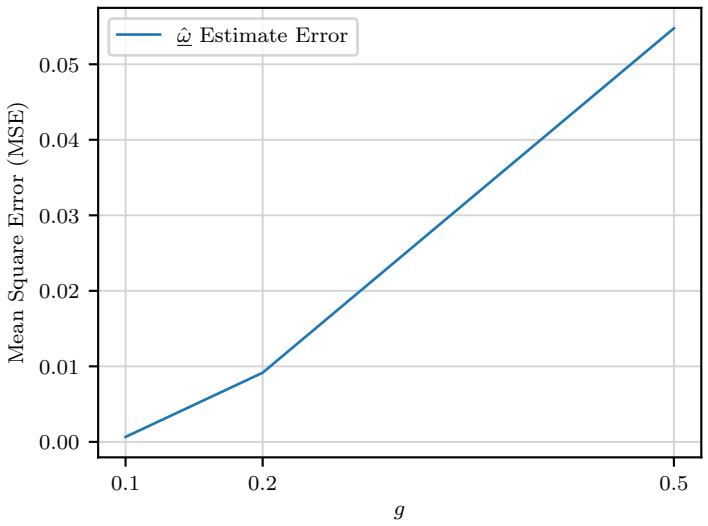


Figure 3.5.: Steady-state MSE of estimated weights $\hat{\underline{\omega}}$ with varying stepsize g.

errors in fusion weights naturally depend on the stepsize g and support the results seen in figure 3.4. Further, an upper bound on this error can be derived by considering the maximum error of each approximation (3.18) and is given by

$$|\hat{\underline{\omega}} - \underline{\omega}| \leq \frac{g}{2}\sqrt{n}\,. \tag{3.27}$$

3.3. Confidential Cloud Fusion Without Leaking Fusion Weights

Previously, we presented a method for solving the estimate fusion problem by weakening the desired cryptographic aims in section 3.1. In this section, we present a method that meets these aims exactly by making a relaxation on the produced fusion output of the fusion cloud.

Broader fusion output In this method, the fusion cloud is not strictly required to produce the fused information vector $\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}}$ and matrix $\mathbf{P}_{\mathsf{fus}}^{-1}$. Instead, any statistics over data from individual sensors i can be computed homomorphically at the fusion cloud and provided to the querying party. For example, the sum statistic over inverted estimate covariance traces, $\sum_{i=1}^{n} \mathrm{tr}(\mathbf{P}_i)^{-1}$, may be returned.

The idea behind the method is to postpone the evaluation of operations that cannot be performed homomorphically until partial fusion results are decrypted by the querying party. The remaining operations are then evaluated on unencrypted inputs to produce the final fusion results. First, we note that FCI fusion (2.38), (2.39) and (2.44) can be rearranged and weights substituted to obtain

$$\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}} = \left(\sum_{i=1}^{n}\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\right)^{-1}\sum_{i=1}^{n}\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\mathbf{P}_i^{-1}\hat{\underline{x}}_i \tag{3.28}$$

and

$$\mathbf{P}_{k,\mathsf{fus}}^{-1} = \left(\sum_{i=1}^{n}\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\right)^{-1}\sum_{i=1}^{n}\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\mathbf{P}_i^{-1}\,. \tag{3.29}$$

In this form, innermost summations

$$\sum_{i=1}^{n}\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\,,\ \sum_{i=1}^{n}\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\mathbf{P}_i^{-1}\ \text{and}\ \sum_{i=1}^{n}\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\mathbf{P}_i^{-1}\hat{\underline{x}}_i \tag{3.30}$$

combine information from individual sensors i, are computable homomorphically given suitable encryption and suit the fusion relaxation defined above. Encryptions of these sums can then be decrypted by the querying party, before remaining inversions and multiplications in (3.28) and (3.29) can be computed to obtain the final results. To depict this straightforward process, pseudocode for the encryption at the sensors, fusion at the cloud and decryption at the querying party are shown in algorithms 1, 2 and 3, respectively. As in the previous section, the Paillier encryption scheme, producing keys pk and sk, is used, encoding from section 2.2.5 is used with $M = N$, where N is the Paillier scheme modulus, and an appropriate precision ϕ is chosen.

Algorithm 1 Encryption at the Sensors

1: **procedure** ESTIMATE(i, pk)
2: Estimate $\mathbf{P}_i^{-1}\hat{\underline{x}}_i$ locally
3: Estimate $\mathbf{P}_i^{-1}$ locally
4: ▷ Encode and encrypt scaling, covariance and estimate components
5: $\xi_i \leftarrow \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\right)\right)$
6: $\underline{b}_i \leftarrow \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\mathbf{P}_i^{-1}\hat{\underline{x}}_i\right)\right)$
7: $\mathbf{B}_i \leftarrow \mathcal{E}_{\mathsf{pk}}\left(\mathsf{E}_0\left(\frac{1}{\mathrm{tr}(\mathbf{P}_i)}\mathbf{P}_i^{-1}\right)\right)$
8: Send ξ_i, $\underline{b}_i$ and $\mathbf{B}_i$ to fusion cloud
9: **end procedure**

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

Algorithm 2 Partial Fusion at the Cloud

procedure PARTIALFUSE(pk)
  Receive ξ_i, $\underline{b}_i$ and $\mathbf{B}_i$ for all $1 \leq i \leq n$
  ▷ Perform homomorphic summations
  $\xi \leftarrow \oplus_{i=1}^{n} \xi_i$
  $\underline{b} \leftarrow \oplus_{i=1}^{n} \underline{b}_i$
  $\mathbf{B} \leftarrow \oplus_{i=1}^{n} \mathbf{B}_i$
  Store ξ, $\underline{b}$ and $\mathbf{B}$ in case of query
end procedure

Algorithm 3 Completing Fusion at the Querying Party

procedure QUERYFUSE(pk, sk)
  Query and receive ξ, $\underline{b}$ and $\mathbf{B}$ from fusion cloud
  ▷ Decrypt and decode
  $\bar{\xi} \leftarrow \mathsf{E}_0^{-1}\left(\mathcal{D}_{\mathsf{pk},\mathsf{sk}}\left(\xi\right)\right)$
  $\bar{\underline{b}} \leftarrow \mathsf{E}_0^{-1}\left(\mathcal{D}_{\mathsf{pk},\mathsf{sk}}\left(\underline{b}\right)\right)$
  $\bar{\mathbf{B}} \leftarrow \mathsf{E}_0^{-1}\left(\mathcal{D}_{\mathsf{pk},\mathsf{sk}}\left(\mathbf{B}\right)\right)$
  ▷ Compute remaining fusion operations
  $\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}} \leftarrow \bar{\xi}^{-1} \cdot \bar{\underline{b}}$
  $\mathbf{P}_{\mathsf{fus}}^{-1} \leftarrow \bar{\xi}^{-1} \cdot \bar{\mathbf{B}}$
  return $\mathbf{P}_{\mathsf{fus}}^{-1}\hat{\underline{x}}_{\mathsf{fus}}$, $\mathbf{P}_{\mathsf{fus}}^{-1}$
end procedure

evaluated sequentially. This can be seen in algorithm 2, where individual components ξ_i, $\underline{b}_i$ and $\mathbf{B}_i$ from each sensor can continue to be sequentially aggregated as sensors send their estimate information. This, in turn, supports the dynamic joining and leaving of sensors in the network without affecting the cloud or the operations of the querying party.

3.3.1. Computational Complexity

As with the previous method, we allow the homomorphic computation of FCI fusion at the computational cost of encryption and homomorphic operations. Here, we look at this additional cost for each involved party using the newly presented scheme. We again assume floating-point and small integer operations to have complexity $O(1)$, the Paillier scheme to have key size $\log N$ bits and repeat the Paillier operation complexities in table 3.3 for convenience. In table 3.4, we apply the Paillier operation complexities to the

Table 3.3.: Computation complexity of Paillier encryption operations.

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

presented method and the unencrypted FCI algorithm is again shown for reference. Here we see that the burden of computation is reduced when compared to the method leaking weights in section 3.2. This can be attributed to no longer requiring the Lewi ORE scheme and the computational simplicity of the final fusion steps at the querying party compared to the required decryption operations. These computational requirements are a necessity for providing the specified security and fusion aims and must be considered when choosing hardware for a physical system where the aims are desired.

3.3.2. Security Analysis

Showing that the cryptographic aims of the fusion problem are met is relatively straightforward. Our aim is for all estimate information, excluding locally produced estimates, that is available to the cloud, eavesdroppers and sensors to be encrypted with a scheme

Table 3.4.: Computation complexity for each party.

	FCI	**Confidential Fusion**
Sensor	$O(1)$	$O\left(d^2 \log^3 N\right)$
Cloud	$O\left(nd^2 + n^3\right)$	$O\left(nd^2 \log^2 N\right)$
Querying Party	$O(1)$	$O\left(d^2 \log^3 N\right)$

meeting the IND-CPA notion. Since the Paillier scheme meets this notion and all transmitted information is encrypted, and noting that the cloud, estimators and eavesdroppers do not hold the secret key sk, the cryptographic aim is met. We do, however, note that the implicit leakage of state dimension d is again present in this method due to the reliance on elementwise encryption.

Lastly, when discussing security, we recall that the partial computation of FCI in this method supports the dynamic joining and leaving of sensors during fusion, but note that the implementation of this in practice introduces additional implicit leakages that need to be considered. Periodic estimation from a sensor may reveal when the sensor is within an estimation range or context, potentially leaking sensor privacy. To combat this, appropriate methods to mitigate implicit leakage need to be considered for real hardware, for example, sending dummy estimate information, $\mathcal{E}_{\mathsf{pk}}(\mathsf{E}_0(0))$, $\mathcal{E}_{\mathsf{pk}}(\mathsf{E}_0(\mathbf{0}))$ and $\mathcal{E}_{\mathsf{pk}}(\mathsf{E}_0(\underline{0}))$, when sensor i is out of estimation context.

3.3.3. Simulation

To demonstrate the accuracy of the method and compare it to the method in section 3.2 as well as the FCI algorithm it approximates, we have implemented a simulation of the fusion scenario. Since errors in fusion are now only introduced during real number quantisation we expect the estimation error to be smaller than that in section 3.2. Code was written in the Python programming language, again using the PHE Paillier encryption scheme library [PythonPaillier2013]. A key size of 512 bits was used, and the constant-velocity linear model in (3.23) was implemented. At each timestep k, the system state $\underline{x}_k$ was measured by $m = 4$ sensors, $1 \leq i \leq 4$, with measurements $\underline{z}_{k,i}$ following the measurement models (3.25) and covariances (3.26). Using a linear IF, initialised with the true state of the system, each sensor produced estimate information $\mathbf{P}_{k,i}^{-1}\hat{\underline{x}}_{k,i}$ and $\mathbf{P}_{k,i}^{-1}$, respectively, that were processed, encrypted and fused at the cloud and querying party. The fusion error results of 1000 simulation runs are shown in figure 3.6. From the figure, we can see the expected similarity in performance between all three methods. The better approximation of the fusion weights in the method from this section results in slightly more accurate results than those with the method from section 3.2, however, choosing a smaller stepsize g can reduce this difference, albeit increase complexity.

3.4. Conclusions on Confidential Estimate Fusion

In this chapter, we have presented two methods for approximating the FCI fusion algorithm on an untrusted cloud where estimates and final fusion meet specified cryptographic aims. The methods primarily differ in whether the FCI fusion weights are leaked to eavesdroppers and the cloud as well as the assumptions on collusions between parties in the network.

The first method, providing confidential fusion with the leakage of weights, introduced in section 3.2, has the benefit of allowing the untrusted cloud to prioritise sensors when performing fusion. This is done by preferring sensors that reduce fused estimate error,

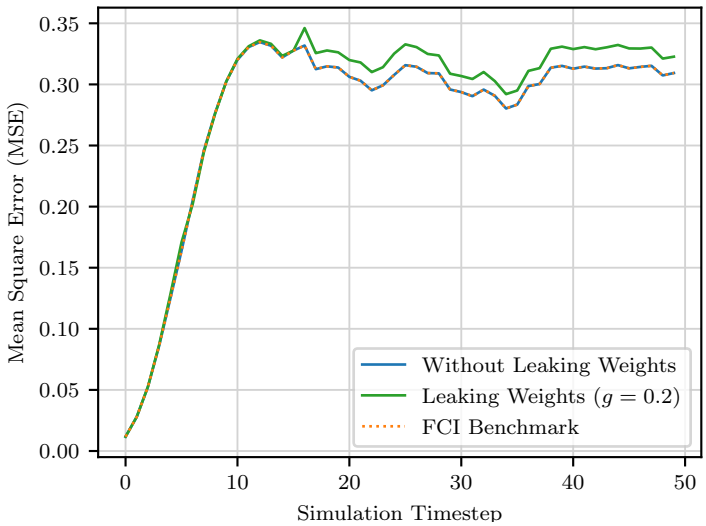


Figure 3.6.: Average MSE of presented fusion methods over 1000 simulations.

as indicated by their fusion weight, but requires the assumption that sensors are trusted and cannot collude maliciously. The method also requires the use of two encryption schemes, resulting in higher computational complexity at sensors and the cloud, as can be seen in section 3.2.3. The second method, presented in section 3.3, provides confidential fusion without the leakage of weights. This method doesn't consider sensors as trusted parties and leaks no information beyond the implicit leakage of state dimension, present in both methods. The stronger security guarantees come with disadvantages, namely that the cloud cannot prioritise estimates based on their fusion error and that an additional, albeit constant, complexity is present at the querying party. Both methods have their computational complexity and security implications analysed and are simulated to demonstrate estimation performance.

Future directions for the topic of confidential data fusion include multi-variable encryption, removing the implicit leakage of state dimension, and the extension of the method to decentralised environments where confidential fusion without a centralised cloud is desirable.

4. Distributed Non-Linear Measurement Fusion with Untrusted Participants

4.1. Problem Formulation

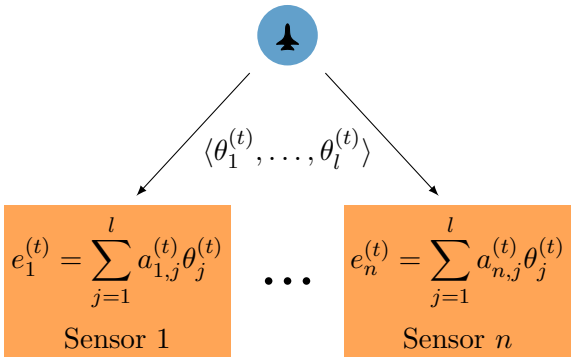
This problem aims to lay down a foundation for solving general non-linear measurement fusion where sensor and navigator privacy is preserved and involved data remains confidential to those producing it. Solving the general problem is made difficult due to the broad measurement definition and the need for concrete communications to be known when proving cryptographic aims, so we first study a specific non-linear problem instead. The presented solution to this problem then lends itself to solving a class of related, but not exhaustive, non-linear measurement fusion problems with the same communication and cryptographic requirements, discussed later in this chapter. We consider the specific context of confidential range sensor navigation, where no sensor is to learn any information about the navigator or other sensors beyond their local measurements, while the navigator learns no 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 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 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.

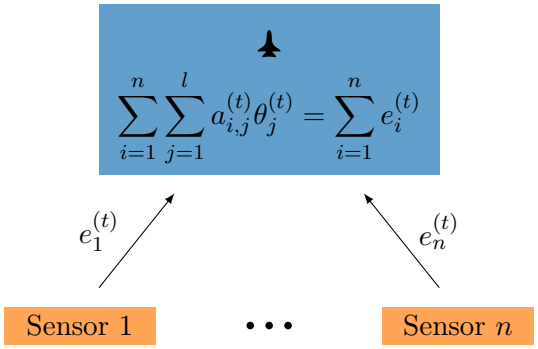
4.1.1. Formal Cryptographic Problem

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 4.3.2, we will show how this protocol is sufficient to compute the location estimate at a navigator while meeting our desired security goals. The communication protocol is as follows.

At every *instance* t (used to distinguish from an estimation *timestep*), the navigator first broadcasts l weights $\theta_j^{(t)}$, $1 \leq j \leq l$, to all sensors i, $1 \leq i \leq n$, who individually compute linear combinations $e_i^{(t)} = \sum_{j=1}^{l} a_{j,i}^{(t)} \theta_i^{(t)}$ based on their measurement data $a_{j,i}$. Linear combinations are then sent back to the navigator, who computes their sum $\sum_{i=1}^{n} e_i^{(t)}$. 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



(a) Broadcast and Combination Step.



(b) Aggregation Step.

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

protocol is computing $\sum_{j=1}^{l}(\theta_j^{(t)} \sum_{i=1}^{n} a_{i,j}^{(t)})$ at the navigator, requiring only values $a_{i,j}^{(t)}$, $1 \leq j \leq l$, 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, $l > n$, and by sending fewer weights in the presence of repeats, as will be shown to be the case in section 4.3.2.

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 chapter.

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 $1 \leq i \leq n$. They are defined as follows.

$\mathsf{Setup}(\kappa)$ On input of security parameter κ, generate public parameters pub, the number of weights l, the navigator's public and private keys $\mathsf{pk_a}$ and $\mathsf{sk_{a,0}}$ and the sensor private keys $\mathsf{sk_{a,}}_i$, $1 \leq i \leq n$.

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

$\mathsf{CombEnc}(t, \mathsf{pk_a}, \mathsf{sk_{a,}}_i, \mathcal{E}_{\mathsf{pk_a}}(\theta_1^{(t)}), \ldots, \mathcal{E}_{\mathsf{pk_a}}(\theta_l^{(t)}), a_{i,1}^{(t)}, \ldots, a_{i,l}^{(t)})$ At instance t, sensor i computes the encrypted linear combination $\mathcal{E}_{\mathsf{pk_a},\mathsf{sk_{a,}}_i}(e_i^{(t)}) = \mathcal{E}_{\mathsf{pk_a},\mathsf{sk_{a,}}_i}(\sum_{j=1}^{l} a_{i,j}^{(t)} \theta_j^{(t)})$ using its secret key $\mathsf{sk_{a,}}_i$.

$\mathsf{AggDec}(t, \mathsf{pk_a}, \mathsf{sk_{a,0}}, e_1^{(t)}, \ldots, e_n^{(t)})$ At instance t, the navigator computes the aggregation of linear combinations $\sum_{i=1}^{n} e_i^{(t)} = \sum_{i=1}^{n} \sum_{j=1}^{l} a_{i,j}^{(t)} \theta_j^{(t)}$ using its public and private keys $\mathsf{pk_a}$, $\mathsf{sk_{a,0}}$.

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 $\theta_j^{(t)}$, $1 \leq j \leq l$, 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, $a_{i,j}^{(t)}$, $1 \leq j \leq l$, where sensor i is not colluding, given only encryptions of their linear combinations e_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, $\sum_{i=1}^{n} e_i^{(t)} = \sum_{i=1}^{n} \sum_{j=1}^{l} a_{i,j}^{(t)} \theta_j^{(t)}$.

While indistinguishable weights can be achieved by encrypting weights with an encryption scheme meeting the IND-CPA notion introduced in section 2.2.1, the novel notion of Linear Combination Aggregator Obliviousness (LCAO) has been formalised as a typical cryptographic game between attacker and challenger in appendix A. 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, $\sum_{i=1}^{n} \sum_{j=1}^{l} a_{i,j}^{(t)} \theta_j^{(t)}$, 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 $(1, 0, \ldots, 0)$, 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 thesis, 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, $1 \leq i \leq n$, 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

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

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 4.3.5, 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. A 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, introduced in section 2.2.2 and 2.2.3, respectively, that provides encrypted weights meeting the IND-CPA notion and encrypted aggregation meeting the LCAO notion in appendix A. 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 $t_{k,\epsilon}$ uniquely capture the scalar aggregation during an estimation timestep k for a single element with position index ϵ. To achieve this in practice, any injective function can be used, such as the concatenation $t_{k,\epsilon} = k \parallel \epsilon$. 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, $l > 1$, and set public parameter $\mathsf{pub} = H$, navigator public key $\mathsf{pk_a} = N$ and navigator private key $\mathsf{sk_{a,0}} = (p, q)$. Sensor secret keys are generated by choosing $\mathsf{sk_{a,}}_i$, $1 \leq i \leq n-1$, uniformly from $\mathbb{Z}_{N^2}$ and setting the last key to $\mathsf{sk_{a,}}_n = -\sum_{i=1}^{n-1} \mathsf{sk_{a,}}_i$.

$\mathsf{Enc}(\mathsf{pk_a}, x)$ Public-key encryption is computed by the Paillier encryption scheme in section 2.2.2. This is given by

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

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

$\mathsf{CombEnc}(t_{k,\epsilon}, \mathsf{pk_a}, \mathsf{sk_{a,}}_i, \mathcal{E}_{\mathsf{pk_a}}(\theta_1^{(k,\epsilon)}), \ldots, \mathcal{E}_{\mathsf{pk_a}}(\theta_l^{(k,\epsilon)}), a_{i,1}^{(k,\epsilon)}, \ldots, a_{i,l}^{(k,\epsilon)})$ At the instance $t_{k,\epsilon}$, encrypted linear combination is given by

$$e_i^{(k,\epsilon)} = H(t_{k,\epsilon})^{\mathsf{sk_{a,}}_i} \prod_{j=1}^{l} \mathcal{E}_{\mathsf{pk_a}}(\theta_j^{(k,\epsilon)})^{a_{i,j}^{(k,\epsilon)}} \pmod{N^2}, \tag{4.7}$$

making use of the Paillier homomorphic properties (2.47) and (2.49). Correctness follows from

$$\begin{aligned}
e_i^{(k,\epsilon)} &= H(t_{k,\epsilon})^{\mathsf{sk_{a,}}_i} \prod_{j=1}^{l} \mathcal{E}_{\mathsf{pk_a}}(\theta_j^{(k,\epsilon)})^{a_{i,j}^{(k,\epsilon)}} \pmod{N^2} \\
&= H(t_{k,\epsilon})^{\mathsf{sk_{a,}}_i} \prod_{j=1}^{l} \mathcal{E}_{\mathsf{pk_a}}(a_{i,j}^{(k,\epsilon)} \theta_j^{(k,\epsilon)}) \pmod{N^2} \\
&= H(t_{k,\epsilon})^{\mathsf{sk_{a,}}_i} \prod_{j=1}^{l} (N+1)^{a_{i,j}^{(k,\epsilon)} \theta_j^{(k,\epsilon)}} \rho_j^N \pmod{N^2} \\
&= H(t_{k,\epsilon})^{\mathsf{sk_{a,}}_i} (N+1)^{\sum_{j=1}^{l} a_{i,j}^{(k,\epsilon)} \theta_j^{(k,\epsilon)}} \tilde{\rho}_i^N \pmod{N^2},
\end{aligned}$$

for some values $\rho_j \in \mathbb{Z}_N$, $1 \leq j \leq l$, and $\tilde{\rho}_i = \prod_{j=1}^{l} \rho_j$. Here, $\tilde{\rho}_i^N$ and $H(t_{k,\epsilon})^{\mathsf{sk_{a,}}_i}$ can be considered the noise terms corresponding to the two levels of encryption from $\mathsf{pk_a}$ and $\mathsf{sk_{a,}}_i$, respectively.

$\mathsf{AggDec}(t_{k,\epsilon}, \mathsf{pk_a}, \mathsf{sk_{a,0}}, e_1^{(k,\epsilon)}, \ldots, e_n^{(k,\epsilon)})$ Aggregation is computed as $e^{(k,\epsilon)} = \prod_{i=1}^{n} e_i^{(k,\epsilon)} \pmod{N^2}$, removing the aggregation noise terms, and is followed by Paillier scheme decryption

$$\sum_{i=1}^{n} \sum_{j=1}^{l} a_{i,j}^{(k,\epsilon)} \theta_j^{(k,\epsilon)} = \frac{L((e^{(k,\epsilon)})^\lambda \pmod{N^2})}{L((N+1)^\lambda \pmod{N^2})} \pmod{N}, \tag{4.8}$$

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

can be seen from

$$\begin{aligned} e^{(k,\epsilon)} &= \prod_{i=1}^{n} H(t_{k,\epsilon})^{\mathsf{sk}_{\mathsf{a},i}} (N+1)^{\sum_{j=1}^{l} a_{i,j}^{(k,\epsilon)} \theta_j^{(k,\epsilon)}} \tilde{\rho}_i^N \pmod{N^2} \\ &= H(t_{k,\epsilon})^{\sum_{i=1}^{n} \mathsf{sk}_{\mathsf{a},i}} \prod_{i=1}^{n} (N+1)^{\sum_{j=1}^{l} a_{i,j}^{(k,\epsilon)} \theta_j^{(k,\epsilon)}} \tilde{\rho}_i^N \pmod{N^2} \\ &= (N+1)^{\sum_{i=1}^{n} \sum_{j=1}^{l} a_{i,j}^{(k,\epsilon)} \theta_j^{(k,\epsilon)}} \tilde{\rho}^N \pmod{N^2}, \end{aligned}$$

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

Additionally, we note that in the above construction, all weights $\theta_j^{(k,\epsilon)}$ 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 appendix B.

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

$$\mathcal{E}_{\mathsf{pk}_\mathsf{a}}(\theta_j^{(k,\epsilon)})^{a_{i,j}^{(k,\epsilon)}} = (N+1)^{\theta_j^{(k,\epsilon)} a_{i,j}^{(k,\epsilon)}} \rho_j^N \pmod{N^2} \tag{4.9}$$

in (4.7), by

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

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 $\theta_j^{(k,\epsilon)}$ that do not hold any information private to the navigator and are known by the sensors in advance.

4.3. Confidential Range-Only Localisation

With a concrete scheme meeting the LCAO notion, we 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 the EIF, introduced in section 2.1.5. The EIF is performed on the information form of the state estimate and its error covariance, repeated here for convenience. That is, the information vector and information matrix,

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

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

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

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 $1 \leq i \leq n$. The updated information vector and matrix can then be used in a local filter prediction step at the navigator, with 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 $\mathbf{H}_{k,i}$. However, computing $\underline{i}_{k,i}$ and $\mathbf{I}_{k,i}$ also requires the current predicted state estimate $\hat{\underline{x}}_{k|k-1}$, when evaluating h_i and $\mathbf{H}_{k,i}$. To achieve the communication protocol desired, we aim to rearrange (4.15) and (4.16) as a linear combination of functions of $\hat{\underline{x}}_{k|k-1}$ (considered the navigator weights), computable at each sensor i, to be subsequently aggregated at the navigator. Application of the linear combination aggregation scheme proposed can then guarantee that sensors do not learn the navigator state, and the navigator learns only the aggregation required for updating its estimate (4.12) and (4.13).

4.3.1. Range Measurement Modification

When rearranging $\underline{i}_{k,i}$ and $\mathbf{I}_{k,i}$ to a linear combination of functions of $\hat{\underline{x}}_{k|k-1}$, we note that h_i does not inherently support this due to the present square-root. Similarly, the

Jacobian of h_i at $\underline{\hat{x}}_{k|k-1}$,

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

does not either. Instead, the modified measurement functions

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

are considered. Now, the functions allow 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 $\hat{y}_{k|k-1}$. For the rearrangement of corresponding modified measurement vectors $\underline{i}'_{k,i}$ and matrices $\mathbf{I}'_{k,i}$, usable in the localisation update step, we also require the existence of measurements following the considered modified measurement models,

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

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 $z_{k,i}$ is complicated by the original noise term $v_{k,i} \sim \mathcal{N}(0, r_{k,i})$. 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. The mean of this new noise term (a function of the Gaussian

term $v_{k,i}$) is given by $\mathsf{E}[2h_i(\underline{x}_k)v_{k,i} + v_{k,i}^2] = r_{k,i}$ and can be used to mean-adjust the squared measurement above, producing the modified measurements

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

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

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

still dependent on h_i. To use the modified measurements (4.23) with the EIF, we require an estimate for $\mathsf{Var}[v'_{k,i}]$ at the sensor as well. Additionally, a conservative estimate, that is, a larger variance resulting in less confidence in measurements, is desirable to reduce filter divergence. Intuitively replacing $h_i(\underline{x}_k)$ with $z_{k,i}$ in (4.24) may not provide a conservative estimate when $z_{k,i}^2 < h_i(\underline{x}_k)^2$, but Gaussianity of $v_{k,i}$ and the squaring of $z_{k,i}$ can be exploited to provide a conservative estimate, with 95% confidence, by adding two of its standard deviations, $\sqrt{r_{k,i}}$, to the replacement term $z_{k,i}$. The modified measurement's variance at timestep k is then 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 variances estimated with (4.25).

4.3.2. Applying the Linear Combination Aggregation Scheme

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 rearranged as

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

with

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

and

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

with

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

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

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

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

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

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, as can be seen from the rearrangements, the number of weights increases combinatorially with the state dimension, thus affecting the cost of communication as well.

4.3.3. Pseudocode

Measurement modification, number encoding and linear combination aggregation are all required to compute the EIF from the previous section and keep all sensor and navigator information confidential. In this section, we summarise this process and give the pseudocode for its execution. As in the previous chapter, we use the Q number format from section 2.2.5 for encoding real number inputs, letting $M = N$, where N is the generated public key, choosing an appropriate precision ϕ and denoting encoding with δ previous multiplications as $\mathsf{E}_\delta(\cdot)$. The confidential localisation filter consists of the following steps.

Setup The Setup algorithm from section 4.2 is run only once by a trusted party. $\mathsf{pub} = H$ and the navigator public key $\mathsf{pk_a} = N$ are made public, and the navigator and sensor secret keys, $\mathsf{sk}_{\mathsf{a},0} = (p, q)$ and $\mathsf{sk}_{\mathsf{a},i}$, $1 \leq i \leq n$, are distributed accordingly.

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

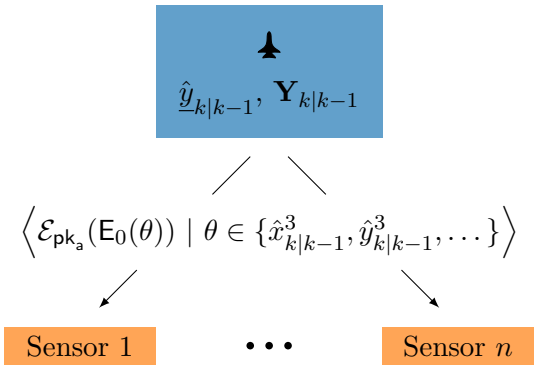
Measurement At each timestep k, sensors modify their measurements with (4.23) and (4.25) before computing elementwise encryptions of $\underline{i}'_{k,i}$ and $\mathbf{I}'_{k,i}$ with CombEnc and sending them back to the navigator. This is shown in algorithm 5.

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

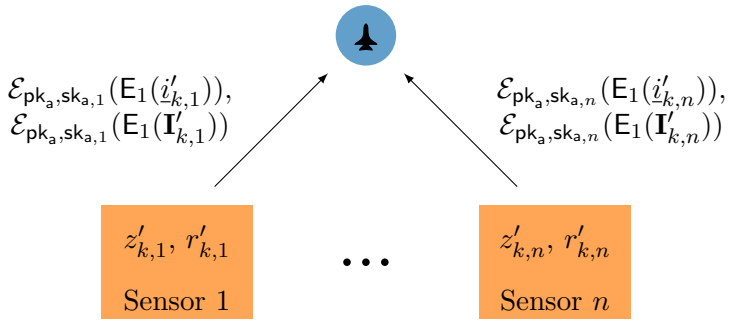
Algorithms 4, 5 and 6 have also been summarised graphically in figure 4.2. Here, $\mathcal{E}_{\mathsf{pk_a},\mathsf{sk}_{\mathsf{a},i}}(\cdot)$ and $\mathsf{E}_\delta(\cdot)$ denote elementwise operations with the same parameters.

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

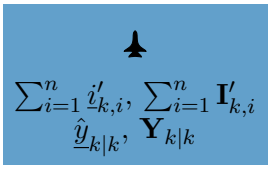
So far in this chapter, we have presented a method for measurement fusion in the context of range-only navigation that meets our desired security goals. The solution can be generalised to solving a sub-class of non-linear measurement problems not limited to range-only navigation and aims to establish the foundations for a general fusion method for non-linear measurements that achieves the same data confidentiality guarantees. Recalling our aim of rewriting (4.15) and (4.16) as a linear combination of functions of $\underline{\hat{x}}_{k|k-1}$, and noting that this was possible when the measurement function h_i could be rewritten in the same way, a more general solution can be seen. That is, *any* non-linear



(a) Prediction.



(b) Measurement.



(c) Update.

Figure 4.2.: Procedure at timestep k for the proposed confidential range-only measurement EIF.

Algorithm 4 Navigator Prediction

procedure PREDICTION($\hat{\underline{y}}_{k-1|k-1}$, $\mathbf{Y}_{k-1|k-1}$, $\mathsf{pk_a}$)
  ▷ Compute local prediction
  Estimate $\hat{\underline{y}}_{k|k-1}$ locally
  Estimate $\mathbf{Y}_{k|k-1}$ locally
  ▷ Encode, encrypt and broadcast weights
  Compute $\mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}^3_{k|k-1}\right)\right)$ given (4.11)
  Broadcast $\mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}^3_{k|k-1}\right)\right)$ to sensors
  for Remaining weights in (4.28) **do**
    Encode, encrypt and broadcast weight in the form above
  end for
  return $\hat{\underline{y}}_{k|k-1}, \mathbf{Y}_{k|k-1}$
end procedure

measurement functions $\underline{h}_{k,i}$ that can be written in the form

$$\underline{h}_{k,i}(\underline{x}) = \sum_{j=1}^{\nu} a_j \underline{\mathcal{H}}_j(\underline{x})\,, \tag{4.31}$$

where all functions $\underline{\mathcal{H}}_j$, $1 \leq j \leq \nu$, do not depend on any sensitive sensor information, are sufficient for rearranging corresponding measurement vectors and matrices, $\underline{i}_{k,i}$ and $\mathbf{I}_{k,i}$, in a similar form and applying the encryption scheme in section 4.2 to the distributed fusion problem.

To stress the applicability of this solution to the sub-class of non-linear problems, we note that the presented method using range-only measurements does not directly fit into this category, as shown in section 4.3.1, requiring modification to measurements to achieve the desired form in (4.31). Similarly, other non-linear measurements that do not directly suit the required form but can be modified accordingly are also solvable by the presented method.

4.3.5. Security Analysis

With the confidential EIF defined in section 4.3.3, we can 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, but recalling that a compromised navigator

Algorithm 5 Measurement at Sensor i

procedure MEASUREMENT(i, $s_{x,i}$, $s_{y,i}$, $r_{k,i}$, pub, $\mathsf{pk_a}$, $\mathsf{sk_{a,}}_i$)
  $H \leftarrow \mathsf{pub}$
  $N \leftarrow \mathsf{pk_a}$
  ▷ Measure and modify measurement
  Measure $z_{k,i}$
  Compute $z'_{k,i}$ by (4.23)
  Compute $r'_{k,i}$ by (4.25)
  ▷ Receive encrypted weights
  Recieve $\mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}^3_{k|k-1}\right)\right)$
  for Remaining weights in (4.28) **do**
    Recieve weight in the form above
  end for
  ▷ Compute linear combination of measurement vector and matrix components
  Let $\boldsymbol{\alpha}_i^{(k,\epsilon)}$ represent the encryption of $\alpha_i^{(k,\epsilon)}$ in (4.26) and (4.27)
  $\boldsymbol{\alpha}_i^{(k,1)} \leftarrow \mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}^3_{k|k-1}\right)\right)^{\mathsf{E}_0\left(2r'^{-1}_{k,i}\right)} \cdot \mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}_{k|k-1}\hat{y}^2_{k|k-1}\right)\right)^{\mathsf{E}_0\left(2r'^{-1}_{k,i}\right)} \cdot$
  $\mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}^2_{k|k-1}\right)\right)^{\mathsf{E}_0\left(-2r'^{-1}_{k,i}s_{x,i}\right)} \cdot \mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{y}^2_{k|k-1}\right)\right)^{\mathsf{E}_0\left(-2r'^{-1}_{k,i}s_{x,i}\right)} \cdot$
  $\mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}_{k|k-1}\right)\right)^{\mathsf{E}_0\left(2r'^{-1}_{k,i}z'_{k,i}\right)} \cdot \mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}_{k|k-1}\right)\right)^{\mathsf{E}_0\left(-2r'^{-1}_{k,i}s^2_{x,i}\right)} \cdot$
  $\mathcal{E}_{\mathsf{pk_a}}\left(\mathsf{E}_0\left(\hat{x}_{k|k-1}\right)\right)^{\mathsf{E}_0\left(-2r'^{-1}_{k,i}s^2_{y,i}\right)} \cdot (N+1)^{\mathsf{E}_1\left(2r'^{-1}_{k,i}s^3_{x,i}\right)} \cdot (N+1)^{\mathsf{E}_1\left(2r'^{-1}_{k,i}s_{x,i}s^2_{y,i}\right)} \cdot$
  $(N+1)^{\mathsf{E}_1\left(-2r'^{-1}_{k,i}s_{x,i}z'_{k,i}\right)} \cdot H(k \parallel 1)^{\mathsf{sk_{a,}}_i} \pmod{N^2}$
  Compute remaining $\boldsymbol{\alpha}_i^{(k,\epsilon)}$ using (4.26), (4.27), (4.7) and the remark from section 4.2 in the form above
  ▷ Send linear combinations to the navigator
  for $\epsilon \leftarrow 1$ to 6 **do**
    Send $\boldsymbol{\alpha}_i^{(k,\epsilon)}$ to the navigator
  end for
end procedure

Algorithm 6 Navigator Update

procedure UPDATE($\underline{\hat{y}}_{k|k-1}$, $\mathbf{Y}_{k|k-1}$, $\mathsf{pk_a}$, $\mathsf{sk_{a,0}}$)
  $N \leftarrow \mathsf{pk_a}$
  ▷ Receive linear combinations from the sensors
  for $\epsilon \leftarrow 1$ to 6 **do**
    Receive $\boldsymbol{\alpha}_i^{(k,\epsilon)}$ from each sensor $1 \leq i \leq n$
  end for
  ▷ Decrypt, decode and construct measurement vector and matrix
  Let $\boldsymbol{\alpha}^{(k,\epsilon)}$ represent an encryption of $\sum_{i=1}^{n} \alpha_i^{(k,\epsilon)}$
  for $\epsilon \leftarrow 1$ to 6 **do**
    $\boldsymbol{\alpha}^{(k,\epsilon)} \leftarrow \prod_{i=1}^{n} \boldsymbol{\alpha}_i^{(k,\epsilon)} \pmod{N^2}$
    Compute $\mathsf{E}_1^{-1}(\mathcal{D}_{\mathsf{pk_a},\mathsf{sk_{a,0}}}(\boldsymbol{\alpha}^{(k,\epsilon)}))$
  end for
  Construct $\sum_{i=1}^{n} \underline{i}'_{k,i}$ and $\sum_{i=1}^{n} \mathbf{I}'_{k,i}$ from decoded decryptions above
  ▷ Perform filter update
  $\underline{\hat{y}}_{k|k} \leftarrow \underline{\hat{y}}_{k|k-1} + \sum_{i=1}^{n} \underline{i}'_{k,i}$
  $\mathbf{Y}_{k|k} \leftarrow \mathbf{Y}_{k|k-1} + \sum_{i=1}^{n} \mathbf{I}'_{k,i}$
  return $\underline{\hat{y}}_{k|k}, \mathbf{Y}_{k|k}$
end procedure

can learn the individual sums weighted by the same weight, the sums

$$\left\{ \sum_{i=1}^{n} 2r_{k,i}^{-1},\ \sum_{i=1}^{n} -r_{k,i}^{-1} s_{x,i},\ \sum_{i=1}^{n} -2r_{k,i}^{-1} s_{x,i},\ \dots \right\}$$

can be leaked as well. From this leakage, we can see that sensitive sensor information, $z'_{k,i}$, $r'_{k,i}$ and $\underline{s}_i$, is present only in their complete sums

$$\sum_{i=1}^{n} z'_{k,i}\,,\ \sum_{i=1}^{n} r'_{k,i}\,,\ \sum_{i=1}^{n} s_{x,i} \text{ and } \sum_{i=1}^{n} s_{y,i}\,, \tag{4.32}$$

which can in practice be interpreted as the leakage of their averages. Therefore, in the context of our proposed localisation method, LCAO leakage corresponds to the averages of sensors' sensitive information, while individual sensor information remains private.

Considering the generalisation of the method discussed in section 4.3.4, the leakage of an LCAO scheme in the context of general measurement fusion can be interpreted similarly. Sensitive sensor information when measurement functions are in the form (4.31) result in only their sums being present in associated leaked measurement vectors and matrix sums, $\sum_{i=1}^{n} \underline{i}_{k,i}$ and $\sum_{i=1}^{n} \mathbf{I}_{k,i}$, corresponding to leakage of average sensitive information only.

4.3.6. 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. As in the previous chapter, a two-dimensional, constant-velocity linear system model,

$$\underline{x}_k = \begin{bmatrix} 1 & 0 & 0.5 & 0 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \underline{x}_{k-1} + \underline{w}_k\,, \tag{4.33}$$

with noise term

$$\underline{w}_k \sim \mathcal{N}\left(\underline{0},\ \frac{1}{10^3} \cdot \begin{bmatrix} 0.42 & 0 & 1.25 & 0 \\ 0 & 0.42 & 0 & 1.25 \\ 1.25 & 0 & 5.0 & 0 \\ 0 & 1.25 & 0 & 5.0 \end{bmatrix} \right), \tag{4.34}$$

was simulated. The navigator used a linear Kalman filter for local state prediction while the algorithms in section 4.3.3 were used for measurement updates. 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 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 timed executions were performed on a 3.33GHz Xeon W3680 CPU, running on the Windows Subsystem for Linux (WSL).

To capture the dependence of the estimated modified measurement variances $r'_{k,i}$ on original measurements $z_{k,i}$, we considered multiple sensor layouts, each with four sensors, with varying average sensor distances from the navigator. The layouts along with the initial state and a sample track are shown visually in figure 4.3. To demonstrate the

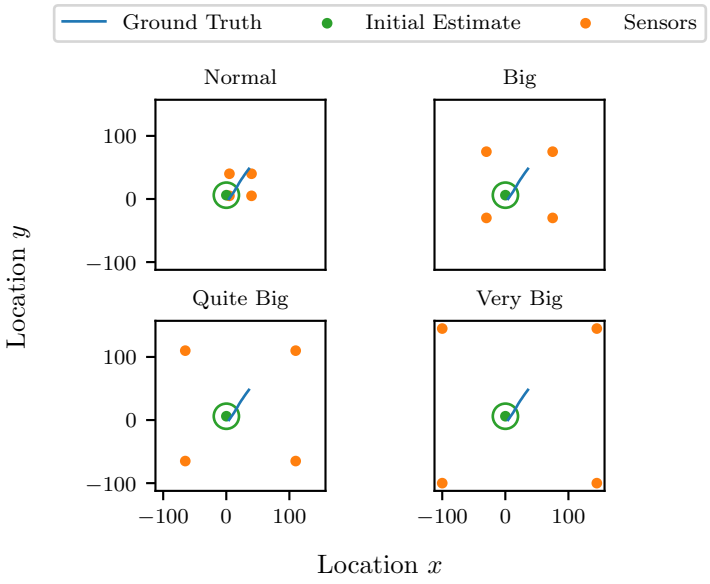


Figure 4.3.: Considered simulation layouts with varying distances between a sample navigator track and sensors.

accuracy of the method, we compared the mean square error (MSE) of the presented filter to the standard EIF using unmodified measurements. Estimation in each layout from figure 4.3 consisted of 50 filter iterations and was run 1000 times. Unmodified measurement variances were chosen as $r_{k,i} = 5$ for all $k > 0$ and a large fractional precision factor, $\phi = 2^{32}$, was chosen. Simulation results can be seen in figure 4.4. From these results, we see the similarity in performance between the presented confidential localisation filter and that of the unmodified EIF. We also see that varying the distances between sensors and the navigator has little impact on the performance of the presented method. We can attribute the similar performance to the conservativeness of estimated modified measurement variances $r'_{k,i}$, eliminating additional filter divergences, and to the high fractional precision factor ϕ, keeping computations consistent with the floating-point arithmetic of the EIF.

In addition to filter performance, computational performance is also an important fac-

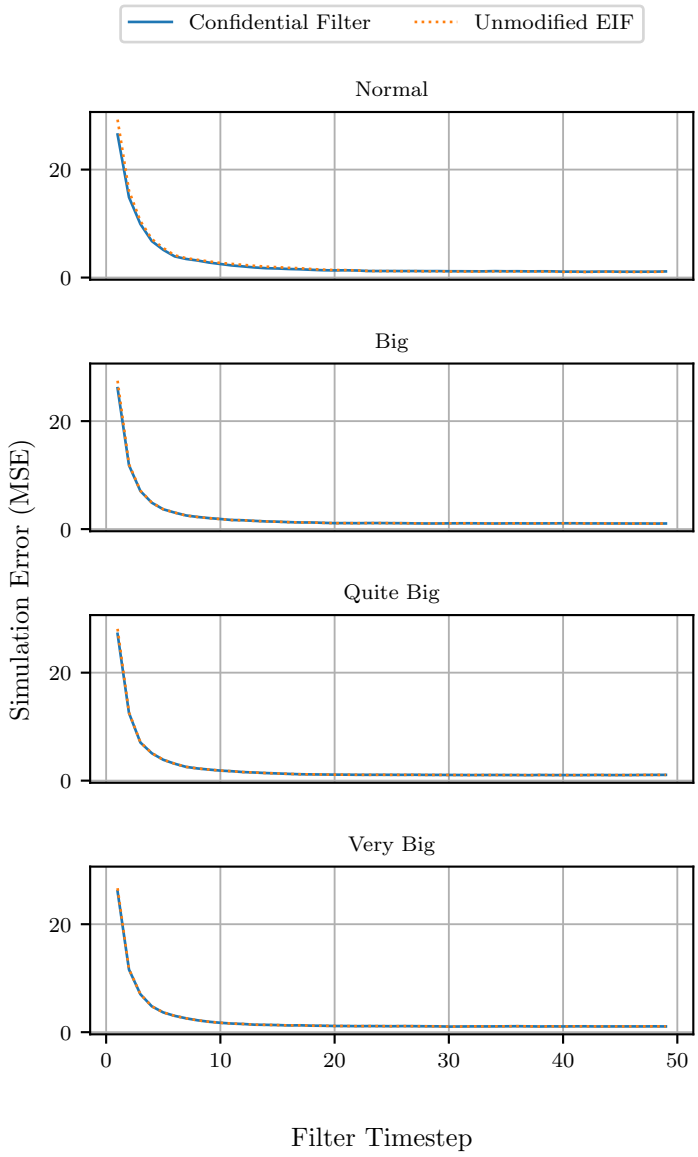


Figure 4.4.: Average MSE of the presented confidential filter for the different layouts.

tor to consider in real-time applications relying on cryptographic methods. Figure 4.5 shows the averages of 10 execution times when varying the numbers of sensors and key sizes (bit lengths of N). Here, increasing the number of sensors primarily affected the number of inter-process communications and aggregation steps due to the asynchronous C implementation. We can see that the predominant computational costs stem from

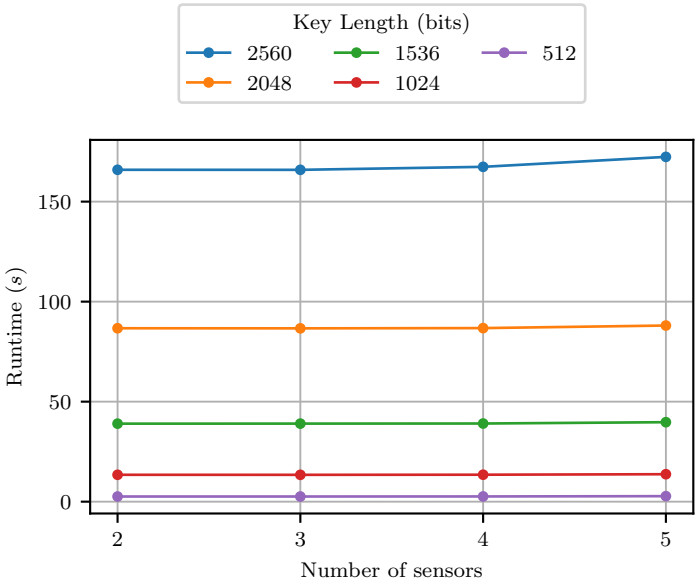


Figure 4.5.: Average simulation runtimes with varying key sizes and numbers of sensors.

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. For example, when relying on the DCRA for security, as is the case for the scheme presented in section 4.2, a key length of 2048 bits is recommended for encrypting government documents [barkerRecommendationPairwiseKey2019]. For our implementation and the aforementioned hardware, a 2048 bit long key results in a filter update roughly every $1.7s$. However, if sensors are mobile and past navigations are not considered confidential, reduced key sizes may be sufficient. Further, a greater decrease in computation time may be achieved with additional code optimisations and more powerful hardware, not considered in this work.

4.4. Conclusions on Confidential Distributed Non-Linear Measurement Fusion

In this chapter, we have presented a localisation filter that, in the presence of range-only sensors, keeps private sensor information and navigator estimates confidential. A suitable cryptographic notion and scheme have been introduced and an implementation of the filter and scheme used to evaluate estimate and computational performance. The

method has additionally been generalised to any non-linear measurement models fitting the form in section 4.3.4 and can find applications in a variety of environments where sensor networks are untrusted or estimates are considered private. Future work on the topic of confidential data fusion includes more computationally efficient schemes meeting the LCAO notion, relaxing the honest-but-curious assumption of malicious sensors and expanding the LCAO notion to enforce the consistent broadcast assumption.

5. Provable Estimation Performances

5.1. Problem Formulation

In this chapter, we look at the problem of formalising estimation performances from a cryptographic perspective and allowing meaningful cryptographic guarantees when comparing estimators. The scenario that we will use to build this formalisation is one where system and measurement models are known and stochastic, and state estimators can have access to secret keys, providing them with a certain privilege. Estimators holding no keys are termed unprivileged. Our goal is to develop a single-sensor scheme that quantifies and cryptographically guarantees a difference between privileged and unprivileged estimator performances when both estimators have access to the same measurements and when models are Gaussian and linear. Further, we look at the extension to multiple sensors and the effect of fusion on cryptographic estimation performance guarantees as well as the applicability of the method to non-linear models.

To capture the aim of comparing a privileged and unprivileged estimator, we first define how to assess the estimation difference between them, and which algorithms are required to characterise a privileged estimation scheme. After giving relevant formal cryptographic definitions, the considered single-sensor privileged estimation problem and its extension to multiple sensors are presented.

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 lends the use of the notion to future literature and is more in line with typical cryptographic practice.

We aim to give the security notion in terms of probabilistic polynomial-time (PPT) attackers and capture the desired leakage as well as attacker capabilities. The most commonly desired leakage, cryptographic indistinguishability, is not suitable for our scenario due to our desire for both estimators to gain *some* information from measurements. Instead, we define security in terms of a time series of semi-definite matrices, given arbitrary known models, such that the difference in estimation error covariances between the estimators with and without access to a privilege, respectively, 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 timestep k denoted by $\underline{x}_k \in \mathbb{R}^d$ and model parameters $\mathcal{M}_{\mathsf{S}}$. Similarly, we assume the existence of a means of process measurement

following a known measurement model exactly, with the measurement at timestep k denoted by $\underline{z}_k \in \mathbb{R}^m$ and model parameters $\mathcal{M}_{\mathsf{M}}$. 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

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

$\mathsf{Noise}(\mathsf{pub}, \mathsf{sk}_{\mathsf{g}}, k, \mathcal{M}_{\mathsf{S}}, \mathcal{M}_{\mathsf{M}}, \underline{z}_1, \ldots, \underline{z}_k)$ On input of public parameters pub, secret key sk_{g}, timestep k, models $\mathcal{M}_{\mathsf{S}}$ and $\mathcal{M}_{\mathsf{M}}$, and measurements $\underline{z}_1, \ldots, \underline{z}_k$, a privileged and unprivileged modified measurement (with no required model constraints) are returned, $\underline{z}_k^{\{\mathsf{p}\}}$ and $\underline{z}_k^{\{\mathsf{up}\}}$, respectively.

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 timestep k.

Definition 5.1.3. A *negligible covariance function*,

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

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 $a \in \mathsf{eig}(\mathbf{A})$, there exists a negligible function [Def. 3.4][katzIntroductionModernCryptography2008] η such that $a \leq \eta(\kappa)$.

Now we can give the security notion that captures the formal requirements of the estimation difference we want to capture.

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

$$\begin{aligned} &\mathsf{Cov}\left[\mathcal{A}\left(k, \kappa, \mathsf{pub}, \mathcal{M}_S, \mathcal{M}_M, \underline{z}_1^{\{\mathsf{up}\}}, \ldots, \underline{z}_k^{\{\mathsf{up}\}}\right) - \underline{x}_k\right] \\ &- \mathsf{Cov}\left[\mathcal{A}'\left(k, \kappa, \mathsf{pub}, \mathcal{M}_S, \mathcal{M}_M, \underline{z}_1^{\{\mathsf{p}\}}, \ldots, \underline{z}_k^{\{\mathsf{p}\}}\right) - \underline{x}_k\right] \\ &\quad \succeq \mathbf{D}_k - \mathsf{negl}\mathsf{Cov}_m(\kappa) \end{aligned} \tag{5.2}$$

for all $k > 0$, some negligible covariance function and where matrices $\mathbf{D}_k$ are semi-definite, *i.e.* $\mathbf{D}_k \preceq 0$ or $\mathbf{D}_k \succeq 0$. 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 $\mathcal{M}_{\mathsf{S}}$ and $\mathcal{M}_{\mathsf{M}}$, estimators $\mathcal{A}$ and $\mathcal{A}'$, and algorithms Setup and Noise.

Informally, the above definition states that no estimator that can only access unprivileged measurements $\underline{z}_1^{\{\mathsf{up}\}}, \ldots, \underline{z}_k^{\{\mathsf{up}\}}$ can estimate a state $\underline{x}_k$ for a timestep k with a

mean square error (MSE) covariance less than an equivalent estimator with access to privileged measurements $\underline{z}_1^{\{\mathsf{p}\}}, \ldots, \underline{z}_k^{\{\mathsf{p}\}}$, 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

To make use of the introduced cryptographic notion, we consider specific estimation models to use in the single-sensor case when developing a privileged estimation scheme with a provable estimation performance difference between privileged and unprivileged estimators. A system model gives the state $\underline{x}_k \in \mathbb{R}^d$ at an integer timestep k and is given by

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

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

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

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

In this scenario, the sensor holds a secret key sk_g that it uses to modify its measurements, and privileged estimators hold this shared key while unprivileged estimators do not. We also assume that sensors and estimators are synchronised in timestep k to simplify later cryptographic evaluation.

5.1.3. Multi-Sensor Problem

As well as the single-sensor problem, we are also interested in the extension to environments with multiple sensors, where the fusion of measurements can also lead to better estimation performance irrespective of privilege. Here, we only consider multiple privileges, such that estimators with a higher privilege should perform better than those with a lower one while taking into consideration the estimation benefits from fusing additional measurements. We again consider linear and Gaussian models, where the state $\underline{x}_k \in \mathbb{R}^d$ follows the system model (5.3). Measurements $\underline{z}_{k,i} \in \mathbb{R}^m$ are now indexed by sensor i, $1 \leq i \leq n$, and follow the measurement models

$$\underline{y}_{k,i} = \mathbf{H}_{k,i} \underline{x}_k + \underline{v}_{k,i} \,, \tag{5.5}$$

with noise terms $\underline{v}_{k,i} \sim \mathcal{N}(\underline{0}, \mathbf{R}_{k,i})$ and known non-zero covariances $\mathbf{R}_{k,i} \in \mathbb{R}^{m \times m}$. In addition to these models, we again assume synchronisation, between all estimators and sensors i, in timesteps k, simplifying later cryptographic evaluation.

Now, each sensor holds its own secret key $\mathsf{sk}_{\mathsf{g},i}$, $1 \leq i \leq n$, which is shared with estimators of appropriate privileges. The privileges that we consider, in terms of access to keys and measurements, will be defined by sequential sensor access. That is, in the

presence of n sensors, we will consider exactly n possible privilege levels, where each privilege $\pi > 0$ corresponds to holding the sequential secret keys $\mathsf{sk}_{\mathsf{g},j}$, $1 \leq j \leq \pi$, while being unprivileged, $\pi = 0$, corresponds to holding none. Additionally, we assume that estimators have access to all privileged measurements, those from sensors whose keys they hold, but can fuse additional unprivileged measurements, from those whose keys they do not hold. To simplify notation, we consider access to unprivileged measurements to be sequential as well, and can therefore capture estimator capabilities by letting $\mathsf{e}^{[\pi,\tau]}$ denote an estimator with privilege π and access to measurements from $\tau \geq \pi$ sensors i, $1 \leq i \leq \tau$.

Multiple measurements and the effects of privilege and fusion on estimation performance complicate the cryptographic analysis in the case of multiple sensors. To demonstrate that a presented scheme guarantees better performance for higher privilege estimators while limiting the benefit from fusing unprivileged measurements, the covariance privilege notion in section 5.1.1 will be used to guarantee two estimation performance differences for each privilege π.

Performance Loss Lower Bound Here, we aim to guarantee a lower bound on the estimation performance loss of any unprivileged estimator $\mathsf{e}^{[0,n]}$ on a privilege-π estimator $\mathsf{e}^{[\pi,\pi]}$. 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 This bound aims to guarantee an upper bound on the estimation performance gain of any estimator $\mathsf{e}^{[\pi,n]}$ on a privilege-π estimator $\mathsf{e}^{[\pi,\pi]}$. The bound similarly remains an upper bound when fewer unprivileged measurements are fused.

Lastly, a suitable scheme should be one with at least two free parameters responsible for controlling 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 $\mathsf{e}^{[\pi',\tau']}$, but are in general beyond the scope of this thesis.

5.2. Privileged Estimation for Linear Systems

In this section, we propose a privileged estimation scheme meeting the security notion in section 5.1.1 for a derivable series of semi-definite matrices when models $\mathcal{M}_{\mathsf{S}}$ and $\mathcal{M}_{\mathsf{M}}$ are given by (5.3) and (5.4), respectively. The key idea behind the method is to add pseudorandom Gaussian noise to existing measurement noise at the sensor, degrading estimation at estimators that cannot remove it. This added noise is a keystream generated by the sensor's secret key and can only be removed from measurements by an estimator holding the same key.

5.2.1. Gaussian Keystream

To generate the desired pseudorandom Gaussian noise that can be added to existing measurements, the sensor first generates a typical cryptographic pseudorandom bitstream with its secret key $\mathsf{sk_g}$. This can be done with any cryptographic stream cipher and reduces the security of the method to a single, well-studied and replaceable component. This bitstream can be interpreted as sequential pseudorandom integers of a suitable size and used to generate a sequence of pseudorandom uniform real numbers $\upsilon_t \dot{\sim} \mathcal{U}(0,1)$ for sequence indices $t > 0$.

Here, we note that the conversion to real numbers υ_t is cryptographically non-trivial due to floating-point representation affecting the pseudorandomness of the samples, and complicating the meeting of a desired cryptographic notion. Instead, we assume that floating-point numbers are sufficiently close to real numbers and rely on any common method for choosing the bit size of pseudorandom integers and the generation of uniform numbers υ_t [goualardGeneratingRandomFloatingPoint2020]. This assumption will be further discussed with the security of the presented scheme in section 5.2.3.

With this assumption, we are left with generating a series of pseudorandom standard normal Gaussian samples, which can be readily computed using the Box-Muller transform [paleyFourierTransformsComplex1934]. This is given by

$$\psi_t = \sqrt{-2\ln(\upsilon_t)}\cos(2\pi\upsilon_{t+1}) \tag{5.6}$$

and

$$\psi_{t+1} = \sqrt{-2\ln(\upsilon_t)}\sin(2\pi\upsilon_{t+1})\,, \tag{5.7}$$

obtaining two, independent, standard normal Gaussian samples from two uniform ones. To generate noise that can be added by the sensor and removed by a privileged estimator using this series, a conversion to a d-dimension zero-mean multivariate Gaussian sample is required at every timestep k. As control over the difference in estimation error between privileged and unprivileged estimators is desired, a symmetric matrix parameter $\mathbf{S} \succ 0$ is introduced, such that added pseudorandom noise $\underline{g}_k$ follows distribution $\underline{g}_k \dot{\sim} \mathcal{N}(\underline{0}, \mathbf{S})$. Given $\mathbf{S}$, $\underline{g}_k$ can be computed using the next d Gaussian keystream samples,

$$\underline{\psi}_k = \begin{bmatrix}\psi_{(k-1)d+1} & \cdots & \psi_{kd}\end{bmatrix}^\top\,, \tag{5.8}$$

as

$$\underline{g}_k = \mathbf{S}^{\frac{1}{2}}\underline{\psi}_k \tag{5.9}$$

for any matrix $\mathbf{S}^{\frac{1}{2}}$ such that $\mathbf{S}^{\frac{1}{2}}\mathbf{S}^{\frac{1}{2}\top} = \mathbf{S}$. We also note that for the correct removal of noise terms $\underline{g}_k$ by the privileged estimator, index information k is required but available when sensors and estimators are synchronised, as per the problem definition.

5.2.2. Measurement Modification

Using the noise in (5.9), the sensor can now modify measurements $\underline{z}_k$ by

$$\underline{z}'_k = \underline{z}_k + \underline{g}_k \,, \tag{5.10}$$

resulting in a new measurement model

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

with noise terms $\underline{v}_k \sim \mathcal{N}(\underline{0}, \mathbf{R}_k)$ and $\underline{g}_k \dot{\sim} \mathcal{N}(\underline{0}, \mathbf{S})$. This leads to two estimation problems for the privileged and unprivileged estimators, respectively.

Privileged estimation An estimator that holds the secret key $\mathsf{sk_g}$ can compute the Gaussian key stream ψ_t, $t > 0$, and therefore the added noise vectors $\underline{g}_k$ at every timestep k. Given the modified measurements (5.10), computing $\underline{z}_k = \underline{z}'_k - \underline{g}_k$ obtains measurements following the measurement model (5.4) exactly.

Unprivileged estimation In the case where pseudorandomness is indistinguishable from randomness, as is the case for an unprivileged estimator when a cryptographically secure keystream is used and the secret key $\mathsf{sk_g}$ is not known, modified measurements are indistinguishable from those following the unprivileged measurement model

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

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

Intuitively, we can see that the two types of estimators have the difference between their estimation errors dependent on matrix $\mathbf{S}$.

5.2.3. Security Analysis

Recalling definition 5.1.4, we aim to show how the notion is met by the proposed 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 in the respective equations in $\mathcal{M}_\mathsf{S}$ and $\mathcal{M}_\mathsf{M}$. Our scheme meets the definition of a formal privileged estimation scheme by defining the required algorithms Setup and Noise as

$\mathsf{Setup}(\mathcal{M}_\mathsf{S}, \mathcal{M}_\mathsf{M}, \kappa)$ Initialize a cryptographically indistinguishable stream cipher with the parameter κ, set the secret key $\mathsf{sk_g}$ to the stream cipher key and include an initial filter estimate $\hat{\underline{x}}_0$, error covariance $\mathbf{P}_0$ and added noise covariance $\mathbf{S}$ in the public parameters pub.

$\mathsf{Noise}(\mathsf{pub}, \mathsf{sk_g}, k, \mathcal{M}_\mathsf{S}, \mathcal{M}_\mathsf{M}, \underline{z}_1, \ldots, \underline{z}_k)$ Using the stream cipher key $\mathsf{sk_g}$ and public parameters pub, create an unprivileged measurement by (5.10). Set and return the privileged measurement $\underline{z}_k^{\{\mathsf{p}\}} = \underline{z}_k$ and unprivileged measurement $\underline{z}_k^{\{\mathsf{up}\}} = \underline{z}'_k$.

Here, we note that in the Setup algorithm above, the inclusion of an initial state estimate, its error covariance and the generated noise covariance in the public parameters pub are present only for the completeness of the cryptographic definition and not a requirement for the security of the scheme.

The idea behind our proof sketch relies on the optimality of the linear Kalman Filter (KF) introduced in section 2.1.2. Given an initial estimate and its error covariance, the KF produces updated estimates with the minimum mean square error (MSE) achievable for *any* estimator when all measurements $\underline{z}_1, \ldots, \underline{z}_k$ are observed, models are Gaussian and linear, and the same initialization is used. Since the KF also preserves the initial error covariance order,

$$\mathbf{P}_k \preceq \mathbf{P}'_k \implies \mathbf{P}_{k+1} \preceq \mathbf{P}'_{k+1}\,, \tag{5.13}$$

for two different filter estimate error covariances $\mathbf{P}_k$ and $\mathbf{P}'_k$, 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 KF error covariance using the combined predict and update equations

$$\begin{aligned}\mathbf{P}_k^{(l)} =& \Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top\big(\mathbf{H}_k(\mathbf{F}_k\mathbf{P}_{k-1}^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top + \mathbf{R}_k\big)^{-1}\mathbf{H}_k\Big)\cdot \\ & \Big(\mathbf{F}_k\mathbf{P}_{k-1}^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k\Big)\,.\end{aligned} \tag{5.14}$$

This gives us a lower bound at every timestep k, such that

$$\mathbf{P}_k^{(l)} \preceq \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_\mathsf{S}, \mathcal{M}_\mathsf{M}, \underline{z}_1, \ldots, \underline{z}_k\right) - \underline{x}_k\right] \tag{5.15}$$

for *any* estimator $\mathcal{A}$ following definition 5.1.2 and any Gaussian and linear models $\mathcal{M}_\mathsf{S}$ and $\mathcal{M}_\mathsf{M}$. This leads us to the proof sketch.

Proof Sketch

We wish to show that the scheme in section 5.2.2 meets $\{\mathbf{D}_1, \mathbf{D}_2, \ldots\}$-Covariance Privilege for Models $\mathcal{M}_\mathsf{S}$ and $\mathcal{M}_\mathsf{M}$, for a computable series $\mathbf{D}_k$, $k > 0$ dependent on a noise parameter $\mathbf{S}$, when $\mathcal{M}_\mathsf{S}$ and $\mathcal{M}_\mathsf{M}$ are Gaussian and linear.

Since a cryptographically pseudorandom stream cipher is used, the stream integers, and therefore the uniform samples υ_t and normal Gaussian samples ψ_t, are indistinguishable from 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 ψ_t are sufficiently close to Gaussian and assume the KF to provide optimal estimation when using floating-point arithmetic. Using the Setup and Noise algorithms given in section 5.2.3 leads to pseudorandom measurements $\underline{z}'_k$ that are indistinguishable from measurements following the unprivileged measurement model (5.12). We can then

compute a lower-bound $\mathbf{P}_k'^{(l)}$ for any unprivileged estimator as $\mathbf{P}_0'^{(l)} = \mathbf{0}$ and

$$\begin{aligned}\mathbf{P}_k'^{(l)} =&\Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}'^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top\big(\mathbf{H}_k(\mathbf{F}_k\mathbf{P}_{k-1}'^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^\top + \mathbf{R}_k + \mathbf{S}\big)^{-1}\mathbf{H}_k\Big)\cdot \\ &\Big(\mathbf{F}_k\mathbf{P}_{k-1}'^{(l)}\mathbf{F}_k^\top + \mathbf{Q}_k\Big)\,.\end{aligned} \tag{5.16}$$

Taking the difference of (5.16) and the lower bound error covariances for privileged estimators (5.14) produces the series

$$\mathbf{D}_k = \mathbf{P}_k'^{(l)} - \mathbf{P}_k^{(l)}\,, \tag{5.17}$$

for $k > 0$, which can be tuned by the parameter $\mathbf{S}$. Since both series $\mathbf{P}_k^{(l)}$ and $\mathbf{P}_k'^{(l)}$ give the lowest possible error covariance of the respective estimators, an estimator following the true model (5.4) can always be created for one following the unprivileged model (5.12) such that their error covariances differ by at least $\mathbf{D}_k$ for each timestep k. A reduction proof can therefore be constructed, in which the existence of an unprivileged estimator that produces estimates such that (5.2) does not hold, implies the existence of an estimator with an error covariance lower than $\mathbf{P}_k'^{(l)}$ following model (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 models are Gaussian and linear, concluding our proof sketch.

Implicit Assumptions

In addition to the proof sketch, we stress some comments on accepting cryptographic guarantees in terms of estimation models $\mathcal{M}_\mathsf{S}$ and $\mathcal{M}_\mathsf{M}$ when used to estimate a physical process or approximate continuous models. The following assumptions are made in this scenario.

Exact models When assigning a model to a physical process, any cryptographic guarantees about the model assumes it describes the process *exactly*. Often, 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 floating-point numbers is beyond the scope of this thesis, their prevalence in the field of state estimation justifies the assumption of sufficient similarity and the insignificance of associated error introduced to the security notion.

Non-Linear Systems

As the presented scheme provides a provable performance difference between privileged and unprivileged estimators when models are Gaussian and linear, it leaves the question of what can be said about the covariance privilege notion in our scheme when models are arbitrary non-linear functions. The basis of our cryptographic guarantee is that optimal estimators for the considered models are known and therefore guarantee a certain difference between privileged and unprivileged estimators' performances. Here, we assume that models are exact but accept that the cryptographic guarantee is useful even when physical processes are not modelled perfectly, as long as optimal linear estimators exist and estimate the process sufficiently well. With this reasoning, we argue that the covariance privilege proof sketch can be similarly applied to non-linear methods when using a non-linear (and non-optimal) estimator. In this case, the difference is no longer cryptographically guaranteed, even if models were exact, since better estimators may exist. However, a derivable difference in performance between known and well-performing estimators, with access to privileged and unprivileged measurements, respectively, still provides meaningful and valuable security information.

5.2.4. Simulation

Simulation results of the presented privileged estimation scheme are shown here in addition to the theoretical backing above. As in previous chapters, we simulated the two-dimension time-invariant constant velocity system model,

$$\underline{x}_k = \begin{bmatrix} 1 & 0 & 0.5 & 0 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \underline{x}_{k-1} + \underline{w}_k \,, \tag{5.18}$$

with noise term

$$\underline{w}_k \sim \mathcal{N}\left(\underline{0}, \begin{bmatrix} 0.42 & 0 & 1.25 & 0 \\ 0 & 0.42 & 0 & 1.25 \\ 1.25 & 0 & 5.0 & 0 \\ 0 & 1.25 & 0 & 5.0 \end{bmatrix} \right) . \tag{5.19}$$

Two measurement models were considered, with bounded and unbounded system errors, respectively, and estimators were implemented using the linear Kalman filter with initial error covariance $\mathbf{0}$. Simulations were written in the Python programming language and the AES block cipher in CTR mode (AES-CTR) [gueronIntelAdvancedEncryption2010] was used as the cryptographically secure stream cipher.

The first measurement model measured state location, leading to an observable system with bounded error covariances as $k \to \infty$. It was given by

$$\mathbf{z}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \underline{x}_k + \underline{v}_k \tag{5.20}$$

and

$$\underline{v}_k \sim \mathcal{N}\left(\underline{0}, \begin{bmatrix} 5 & 2 \\ 2 & 5 \end{bmatrix}\right) . \tag{5.21}$$

The sensor added pseudorandom Gaussian samples with a covariance $\mathbf{S} = 35 \cdot \mathbf{I}$ according to our scheme in section 5.2.2. Figure 5.1 shows the average error covariance traces and the mean square error (MSE) of a privileged and unprivileged estimator for 1000 simulations runs using the models (5.18) and (5.20). As expected, it can be seen that the

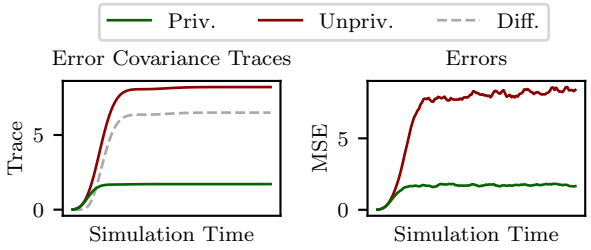


Figure 5.1.: Privileged estimation with bounded error covariance.

privileged estimator's error covariance trace is lower than the unprivileged estimator's and that the privileged estimator has a lower MSE. The difference in trace between the two estimators has also been plotted and equals the trace of the series (5.17) due to the simulation initial error covariance $\mathbf{0}$.

The second simulation considers an unobservable system where only state velocity is measured and has an unbounded error covariance as $k \to \infty$. It is given by

$$\mathbf{z}_k = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \underline{x}_k + \underline{v}_k \,, \tag{5.22}$$

and the same noise distribution and keystream covariance $\mathbf{S}$ as in the bounded case. Figure 5.2 shows the average error covariance traces and MSE of estimation from 1000 simulation runs with models (5.18) and (5.22) and shows similar results.

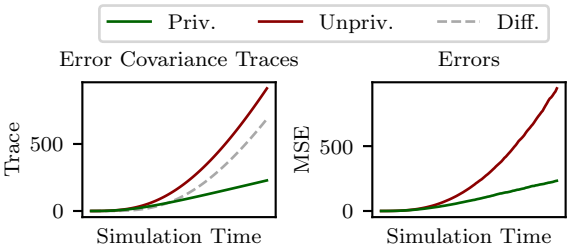


Figure 5.2.: Privileged estimation with unbounded error covariance.

Both figures capture the difference in estimation error between the best possible estimators given the simulated processes (in terms of MSE) and support the security proof sketch in section 5.2.3.

5.3. Fusion in Privileged Estimation Environments

Recalling the problem formulation in section 5.1.3, an effective single-sensor privileged estimation scheme leads to interest in the extension to environments with multiple sensors, where multiple privileges and accesses to measurements are possible, affecting the estimation performance of present estimators. We aim to present a privileged fusion scheme where the Performance Loss Lower Bound (PLLB) and the Performance Gain Upper Bound (PGUB), defined in section 5.1.3, can be derived and proved using the covariance privilege notation in definition 5.1.4. The idea behind the 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. Correlated Gaussian Keystreams

Similarly to the multivariate Gaussian keystream in section 5.2.1, 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 $\mathbf{V} \in \mathbb{R}^{m\times m}$, $\mathbf{V} \succ \mathbf{0}$, and an uncorrelated component $\mathbf{W} \in \mathbb{R}^{m\times m}$, $\mathbf{W} \succ \mathbf{0}$, and define a noise cross-correlation matrix for x noises as $\mathbf{S}^{(x)} \in \mathbb{R}^{xm\times xm}$,

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

and $\mathbf{S}^{(1)} = \mathbf{V} + \mathbf{W}$. Denoting generated multivariate standard Gaussian noise (5.8) and added Gaussian noise (5.9) for sensor i at timestep k as $\underline{\psi}_{k,i}$ and $\underline{g}_{k,i}$, respectively, the generation of all n multivariate Gaussian noises at timestep k, $\underline{g}_k^{(1:n)}$, can be computed. This can be done by

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

where each $\underline{\psi}_{k,i}$ is computed as $\underline{\psi}_k$ in (5.8) using uniform samples generated with key $\mathsf{sk}_{\mathsf{g},i}$, 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, as we consider sequential access to keys, it is important that the vector of the first π noises $\underline{g}_{k,i}$, $1 \leq i \leq \pi$, in (5.24), denoted $\underline{g}_k^{(1:\pi)}$, can be reproduced by an estimator of privilege π, holding only the keys $\mathsf{sk}_{\mathsf{g},i}$, $1 \leq i \leq \pi$. 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 $\mathbf{S}^{(n)}$. Then, each correlated Gaussian sample $\underline{g}_{k,i}$ is computable from preceding standard samples $\underline{\psi}_{k,j}$,

$j \leq i$ only, and the generalised noise generation equation

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

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

At every timestep k, $\underline{g}_k^{(1:n)}$ can be generated with (5.25) using all n keys and used to modify sensor measurements, while the subset $\underline{g}_k^{(1:\pi)}$ can be generated by estimators of privilege π 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 in the multiple-sensor environment. Measurement modification is performed 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,

$$\underline{z}'_{k,i} = \underline{z}_{k,i} + \underline{g}_{k,i} = \mathbf{H}_{k,i}\underline{x}_k + \underline{v}_{k,i} + \underline{g}_{k,i}\,, \tag{5.26}$$

with measurement noise $\underline{v}_{k,i} \sim \mathcal{N}(\underline{0}, \mathbf{R}_{k,i})$ and the vector of all added pseudorandom noises $\underline{g}_k^{(1:n)} \mathrel{\dot\sim} \mathcal{N}(\underline{0}, \mathbf{S}^{(n)})$. 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 τ measurements at timestep k, as

$$\underline{z}'^{(1:\tau)}_k = \underline{z}^{(1:\tau)}_k + \underline{g}^{(1:\tau)}_k = \mathbf{H}^{(1:\tau)}_k \underline{x}_k + \underline{v}^{(1:\tau)}_k + \underline{g}^{(1:\tau)}_k\,, \tag{5.27}$$

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

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

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

and $\mathbf{S}^{(\tau)} \in \mathbb{R}^{\tau m \times \tau m}$ 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. Recalling the estimator notation introduced in section 5.1.3, we have

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

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

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

Estimators of the form $\mathsf{e}^{[\pi,\pi]}$ 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, $\underline{g}_k^{(1:\pi)}$ can be generated and $\underline{z}_k^{[\pi,\pi]} = \underline{z}_k'^{(1:\pi)} - \underline{g}_k^{(1:\pi)}$ computed to give the observed measurement model equal to receiving unmodified measurements only,

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

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

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

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

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

$$\underline{g}_k^{(\pi+1:\tau)} \mid \underline{g}_k^{(1:\pi)} \mathrel{\dot\sim} \mathcal{N}\left(\bar{\mathbf{V}}^\top \mathbf{S}^{(\pi)-1} \underline{g}_k^{(1:\pi)}, \mathbf{S}^{(\tau-\pi)} - \bar{\mathbf{V}}^\top \mathbf{S}^{(\pi)-1} \bar{\mathbf{V}} \right) . \tag{5.31}$$

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

$$\underline{z}_k^{[\pi,\tau]} = \underline{z}_k'^{(1:\tau)} - \begin{bmatrix} \underline{g}_k^{(1:\pi)} \\ \bar{\mathbf{V}}^\top \mathbf{S}^{(\pi)-1} \underline{g}_k^{(1:\pi)} \end{bmatrix} , \tag{5.32}$$

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

$$\underline{z}_k^{[\pi,\tau]} = \mathbf{H}_k^{(1:\tau)}\underline{x}_k + \underline{v}'_k \tag{5.33}$$

where

$$\underline{v}'_k \sim \mathcal{N}\left(\underline{0}, \begin{bmatrix} \mathbf{R}_k^{(1:\pi)} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}^{(\tau-\pi)} - \bar{\mathbf{V}}^\top \mathbf{S}^{(\pi)-1}\bar{\mathbf{V}} + \mathbf{R}_k^{(\pi+1:\tau)} \end{bmatrix}\right). \tag{5.34}$$

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 $\tau - \pi$ non-sequential unprivileged measurements $\underline{z}_{k,i}$, $\pi < i \leq \tau$, by appropriately rearranging the columns of $\mathbf{S}^{(\tau-\pi)}$ in (5.30).

From the observed measurement models (5.28), (5.29) and (5.33) we can tell that the parameters $\mathbf{V}$ and $\mathbf{W}$ (within matrices $\mathbf{S}^{(\tau)}$, $\mathbf{S}^{(\pi)}$ and $\mathbf{S}^{(\tau-\pi)}$) will control the difference in estimation performance between the three types of estimators. Computing the two bounds we wish to cryptographically guarantee, PLLB and PGUB, respectively, and how $\mathbf{V}$ and $\mathbf{W}$ 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, (**??**) 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 (**??**).

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 (**??**), as

$$\begin{aligned} \underline{g}_{k,i} =& \bar{\mathbf{Z}}^\top \mathbf{S}^{(i-1)-1}\underline{g}_k^{(1:i-1)} + \\ & (\mathbf{S}^{(1)} - \bar{\mathbf{Z}}^\top \mathbf{S}^{(i-1)-1}\bar{\mathbf{Z}})^{\frac{1}{2}}\underline{z}_{k,i}\,. \end{aligned} \tag{5.35}$$

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 (**??**), (**??**) and (**??**) 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, (**??**) and (**??**), 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 [eqn:system_model], all measurements models [eqn:measurement_models] (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 $\hat{\underline{x}}_0$ 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 (**??**) and $\underline{y}_k^{\{\mathsf{p}\}} = \underline{y}_k^{[p,p]}$ following (**??**).

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

$$\begin{aligned}\mathbf{P}_k^{[0,N]} =& \Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}^{[0,N]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top}\cdot \\ &\quad \big(\mathbf{H}_k^{(1:N)}(\mathbf{F}_k\mathbf{P}_{k-1}^{[0,N]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top} \\ &\quad + \mathbf{R}_k^{(1:N)} + \mathbf{S}^{(N)}\big)^{-1}\mathbf{H}_k^{(1:N)}\Big)\cdot \\ &\Big(\mathbf{F}_k\mathbf{P}_{k-1}^{[0,N]}\mathbf{F}_k^\top + \mathbf{Q}_k\Big),\end{aligned} \tag{5.36}$$

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

$$\begin{aligned}\mathbf{P}_k^{[p,p]} =& \Big(\mathbf{I} - (\mathbf{F}_k\mathbf{P}_{k-1}^{[p,p]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:p)\top}\cdot \\ &\quad \big(\mathbf{H}_k^{(1:p)}(\mathbf{F}_k\mathbf{P}_{k-1}^{[p,p]}\mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:p)\top} \\ &\quad + \mathbf{R}_k^{(1:p)}\big)^{-1}\mathbf{H}_k^{(1:p)}\Big)\cdot \\ &\Big(\mathbf{F}_k\mathbf{P}_{k-1}^{[p,p]}\mathbf{F}_k^\top + \mathbf{Q}_k\Big)\end{aligned} \tag{5.37}$$

and $\mathbf{P}_0^{[p,p]} = \mathbf{0}$. The bounds (5.36) and (5.37) are constructed such that at every timestep k,

$$\begin{aligned}&\mathbf{P}_k^{[0,N]} \preceq \\ &\quad \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1^{[0,N]}, \ldots, \underline{y}_k^{[0,N]}\right) - \underline{x}_k\right]\end{aligned} \tag{5.38}$$

and

$$\begin{aligned}&\mathbf{P}_k^{[p,p]} \preceq \\ &\quad \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1^{[p,p]}, \ldots, \underline{y}_k^{[p,p]}\right) - \underline{x}_k\right]\end{aligned} \tag{5.39}$$

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

$$\mathbf{D}_{\mathsf{PLLB},k} = \mathbf{P}_k^{[0,N]} - \mathbf{P}_k^{[p,p]} \tag{5.40}$$

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 (5.37), 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 [eqn:system_model] and Stacked Measurement Models [eqn:measurement_models].*

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 [subsec:crypto_performance_loss_lower_bound] 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 (**??**) and $\underline{y}_k^{\{\mathsf{p}\}} = \underline{y}_k^{[p,p]}$ following (**??**).

The minimum error covariances achievable by an estimator $\mathsf{e}^{[p,p]}$ is again given by (5.37) 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

$$\begin{aligned}\mathbf{P}_k^{[p,N]} =& \Big(\mathbf{I} - (\mathbf{F}_k \mathbf{P}_{k-1}^{[p,N]} \mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top} \cdot \\ & \big(\mathbf{H}_k^{(1:N)} (\mathbf{F}_k \mathbf{P}_{k-1}^{[p,N]} \mathbf{F}_k^\top + \mathbf{Q}_k)\mathbf{H}_k^{(1:N)\top} \\ & + \mathbf{X}\big)^{-1} \mathbf{H}_k^{(1:N)}\Big)\Big(\mathbf{F}_k \mathbf{P}_{k-1}^{[p,N]} \mathbf{F}_k^\top + \mathbf{Q}_k\Big),\end{aligned} \tag{5.41}$$

where

$$\mathbf{X} = \begin{bmatrix} \mathbf{R}_k^{(1:p)} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}^{(N-p)} - \bar{\mathbf{Z}}^\top \mathbf{S}^{(p)-1} \bar{\mathbf{Z}} + \mathbf{R}_k^{(p+1:N)} \end{bmatrix} \tag{5.42}$$

and $\mathbf{P}_0^{[p,N]} = \mathbf{0}$. Again, the bounding series are such that (5.39) and

$$\begin{aligned}&\mathbf{P}_k^{[p,N]} \preceq \\ &\quad \mathsf{Cov}\left[\mathcal{A}\left(k, \mathcal{M}_S, \mathcal{M}_M, \underline{y}_1^{[p,N]}, \dots, \underline{y}_k^{[p,N]}\right) - \underline{x}_k\right]\end{aligned} \tag{5.43}$$

hold. Now, the difference

$$\mathbf{D}_{\mathsf{PGUB},k} = \mathbf{P}_k^{[p,N]} - \mathbf{P}_k^{[p,p]} \tag{5.44}$$

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 (5.37), 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 [eqn:system_model] and Stacked Measurement Models [eqn:measurement_models].*

In (5.44), $\mathbf{D}_{\mathsf{PGUB},k} \preceq 0$ 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 [eqn:system_model] with parameters

$$\mathbf{F}_k = \begin{bmatrix} 1 & 0.5 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.45}$$

and

$$\mathbf{Q}_k = 10^{-3} \cdot \begin{bmatrix} 0.42 & 1.25 & 0 & 0 \\ 1.25 & 5 & 0 & 0 \\ 0 & 0 & 0.42 & 1.25 \\ 0 & 0 & 1.25 & 5 \end{bmatrix}, \tag{5.46}$$

for all k, and measured independently by location sensors i, $1 \leq i \leq N = 4$, following [eqn:measurement_models] with constant parameters

$$\mathbf{H}_{k,i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \text{ and } \mathbf{R}_{k,i} = \begin{bmatrix} 5 & 2 \\ 2 & 5 \end{bmatrix}, \tag{5.47}$$

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 5.3 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 (5.40) and PGUB series (5.44), respectively, when $\Sigma_z = 2$ and $\Sigma_y = 10$.

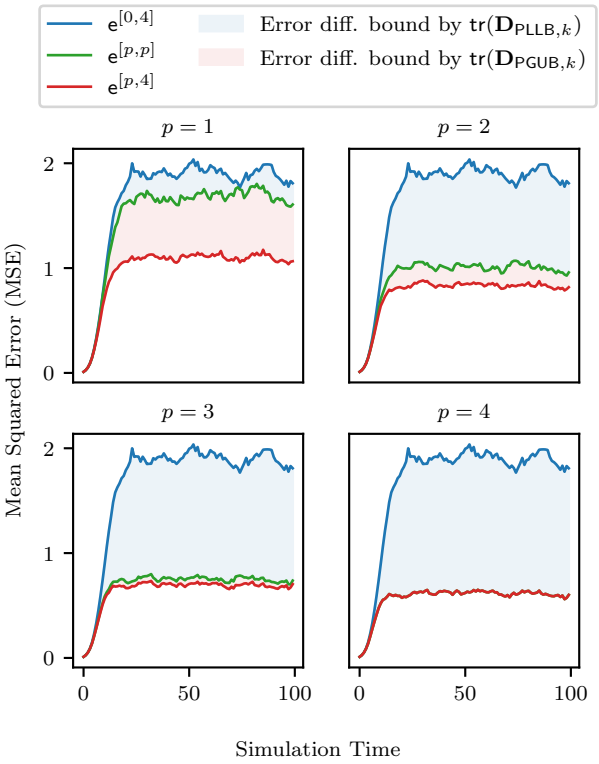


Figure 5.3.: The average errors of different estimators for 1000 simulation runs when $\Sigma_z = 2$ and $\Sigma_y = 10$.

To demonstrate the effect of parameters Σ_z and Σ_y (and therefore $\mathbf{Z}$ and $\mathbf{Y}$), figure 5.4 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 5.5 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 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-

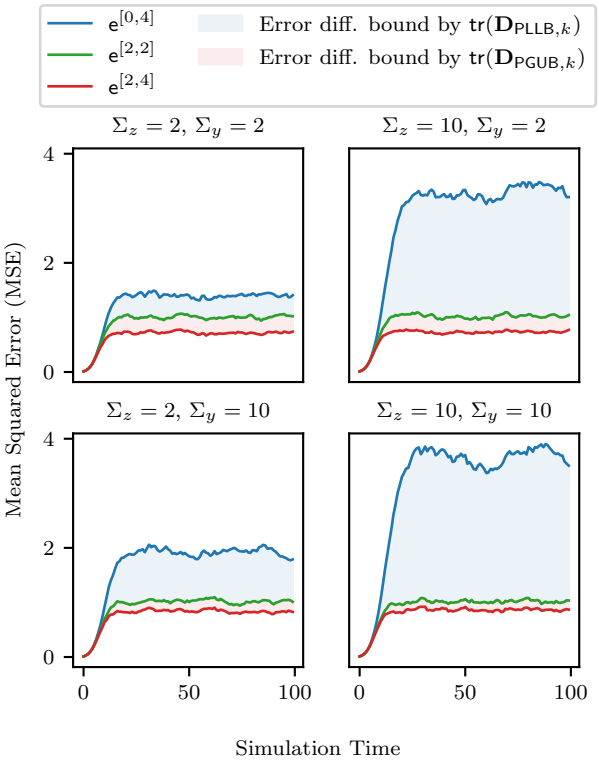


Figure 5.4.: The average errors of unprivileged and privilege-2 estimators for 1000 simulation runs when varying Σ_z and Σ_y.

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 5.5 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 $\mathsf{tr}(\mathbf{D}_{\mathsf{PLLB},k})$).

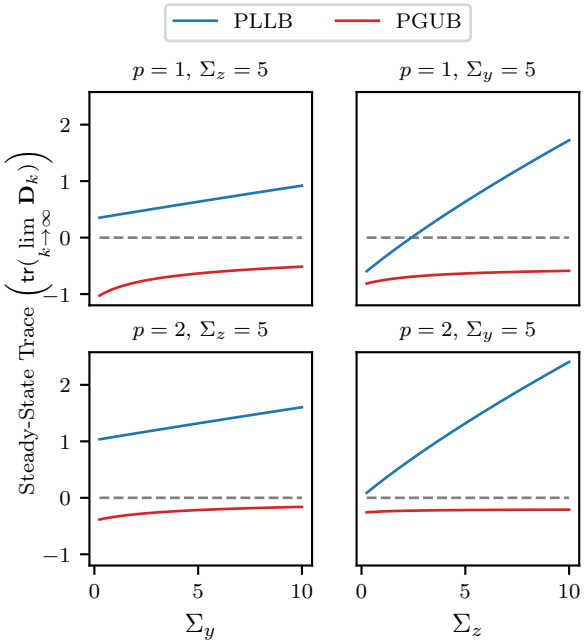


Figure 5.5.: Steady-state traces of the PLLB and PGUB for privileges $p = 1$ and $p = 2$ when Σ_z and Σ_y are varied.

5.4. Conclusions on Provable Estimation Performances

6. Conclusion

A. Linear-Combination Aggregator Obliviousness (LCAO)

The following game between attacker and challenger defines the security notion of LCAO.

Setup The challenger chooses security parameter κ, runs the $\mathsf{Setup}(\kappa)$ algorithm and gives pub, l and $\mathsf{pk_a}$ to the attacker

Queries The attacker can now perform encryptions or submit queries that are answered by the challenger. The types of actions are:

1. *Encryption:* The attacker chooses a value x and computes an encryption of x under the aggregator's public key $\mathsf{pk_a}$, obtaining $\mathcal{E}_{\mathsf{pk_a}}(x)$.
2. *Weight Queries:* The attacker chooses an instance t and receives the weights for that instance encrypted with the aggregator's public key, $\mathcal{E}_{\mathsf{pk_a}}(\theta_j^{(t)})$, $1 \leq j \leq l$.
3. *Combine Queries:* The attacker chooses a tuple $(i, t, a_{i,1}^{(t)}, \ldots, a_{i,l}^{(t)})$ such that for any two chosen query tuples $(i, t, a_{i,1}^{(t)}, \ldots, a_{i,l}^{(t)})$ and $(i', t', a_{i',1}^{\prime(t')}, \ldots, a_{i',l}^{\prime(t')})$, the following condition holds:

$$i = i' \wedge t = t' \implies a_{i,j}^{(t)} = a_{i',j}^{\prime(t')},\ 1 \leq j \leq l\,.$$

   The attacker is then given back the encryption of the linear combination $\mathcal{E}_{\mathsf{pk_a},\mathsf{sk}_{\mathsf{a},i}}(\sum_{j=1}^{l} a_{i,j}^{(t)}\theta_j^{(t)})$ encrypted under both the aggregator public key $\mathsf{pk_a}$ and the secret key $\mathsf{sk}_{\mathsf{a},i}$.
4. *Compromise queries:* The attacker chooses i and receives the secret key $\mathsf{sk}_{\mathsf{a},i}$. The aggregator's secret key may also be compromised (when choosing $i = 0$).

Challenge Next, the attacker chooses an instance t^*, and a subset of users $S \subseteq U$ where U is the complete set of users for which no combine queries, for the instance t^*, and no compromise queries, are made for the duration of the game. The attacker then chooses two series of tuples

$$\left\langle \left(i, t^*, a_{i,1}^{(t^*)(0)}, \ldots, a_{i,l}^{(t^*)(0)}\right) \;\middle|\; i \in S \right\rangle$$

and

$$\left\langle \left(i, t^*, a_{i,1}^{(t^*)(1)}, \ldots, a_{i,l}^{(t^*)(1)}\right) \;\middle|\; i \in S \right\rangle,$$

and gives them to the challenger. In the case that $0 \in S$ (*i.e.*, the aggregator is compromised) and $S = U$, it is additionally required that

$$\sum_{i \in S} \sum_{j=1}^{l} a_{i,j}^{(t^*)(0)} \theta_j^{(t^*)} = \sum_{i \in S} \sum_{j=1}^{l} a_{i,j}^{(t^*)(1)} \theta_j^{(t^*)} ,$$

for weights $\theta_j^{(t^*)}$, $1 \leq j \leq l$, returned by a *Weight Query* with chosen instance t^*. The challenger then chooses a random bit $\beta \in \{1, 0\}$ and returns encryptions

$$\left\langle \mathcal{E}_{\mathsf{pk_a},\mathsf{sk_a},i} \left(\sum_{j=1}^{l} a_{i,j}^{(t^*)(\beta)} \theta_j^{(t^*)} \right) \;\middle|\; i \in S \right\rangle .$$

More Queries The attacker can now perform more encryptions and submit queries, so long as the queries do not break the requirements in the Challenge stage. That is, $S \subseteq U$.

Guess At the end, the attacker outputs a bit β' and wins the game if and only if $\beta' = \beta$. The advantage of an attacker $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{LCAO}(\mathcal{A}) \coloneqq \left| \mathrm{P}[\beta' = \beta] - \frac{1}{2} \right| .$$

Definition A.0.1. An encryption scheme meets LCAO security if no probabilistic adversary, running in polynomial-time with respect to the security parameter κ, has more than a negligible advantage in winning the above security game. That is, for all adversaries $\mathcal{A}$, there exists a negligible function η, such that

$$\mathsf{Adv}^{LCAO}(\mathcal{A}) \leq \eta(\kappa) ,$$

with probabilities taken over randomness introduced by $\mathcal{A}$, and in Setup, Enc and $\mathsf{CombEnc}$.

B. Cryptographic Proof for Meeting the LCAO Notion

The scheme in section 4.2 will be shown to meet LCAO by contrapositive. We show that for any adversary $\mathcal{A}$ playing against a challenger using the scheme, we can always create an adversary $\mathcal{A}'$ playing against a challenger $\mathcal{C}$ using the Joye-Libert scheme, such that

$$\mathsf{Adv}^{LCAO}(\mathcal{A}) > \eta_1(\kappa) \implies \mathsf{Adv}^{AO}(\mathcal{A}') > \eta_2(\kappa)\,,$$

for *any* negligible functions η_1, η_2 and security parameter κ. That is, if we assume our scheme does not meet LCAO, then the Joye-Libert scheme in section 2.2.3 does not meet AO (which is not the case, [joyeScalableSchemePrivacyPreserving2013]).

Proof. Consider adversary $\mathcal{A}$ playing the LCAO game. The following is a construction of an adversary $\mathcal{A}'$ playing the AO game [shiPrivacyPreservingAggregationTimeSeries2011] against a challenger $\mathcal{C}$ using the Joye-Libert aggregation scheme.

Setup When receiving N and H as public parameters from $\mathcal{C}$, choose an $l > 1$ and give public parameter H, number of weights l, and $\mathsf{pk_a} = N$ to $\mathcal{A}$.

Queries Handle queries from $\mathcal{A}$:

Weight Query When $\mathcal{A}$ submits a weight query t, choose weights $\theta_j^{(t)}$, $1 \le j \le l$, and random values $\rho_j \in \mathbb{Z}_N$, $1 \le j \le l$, and return encryptions

$$(N+1)^{\theta_j^{(t)}} \rho_j^N \pmod{N^2},\ 1 \le j \le l\,,$$

to $\mathcal{A}$.

Combine Query When $\mathcal{A}$ submits a combine query $(i, t, a_{i,1}^{(t)}, \ldots, a_{i,l}^{(t)})$, choose the weights $\theta_j^{(t)}$, $1 \le j \le l$, if not already chosen for the instance t, and make an AO encryption query $(i, t, \sum_{j=1}^{l} a_{i,j}^{(t)} \theta_j^{(t)})$ to $\mathcal{C}$. The received response will be of the form $(N+1)^{\sum_{j=1}^{l} a_{i,j}^{(t)} \theta_j^{(t)}} H(t)^{\mathsf{sk}_{\mathsf{a},i}}$; multiply it by $\tilde{\rho}^N$ for a random $\tilde{\rho} \in \mathbb{Z}_N$ and return

$$(N+1)^{\sum_{j=1}^{l} a_{i,j}^{(t)} \theta_j^{(t)}} \tilde{\rho}^N H(t)^{\mathsf{sk}_{\mathsf{a},i}} \pmod{N^2}$$

to $\mathcal{A}$.

Compromise Query When $\mathcal{A}$ submits compromise query i, make the same compromise query i to $\mathcal{C}$, and return the recieved secret key $\mathsf{sk}_{\mathsf{a},i}$ to $\mathcal{A}$.

Challenge When $\mathcal{A}$ submits challenge series

$$\left\langle \left(i, t^*, a_{i,1}^{(t^*)(0)}, \ldots, a_{i,l}^{(t^*)(0)}\right) \;\middle|\; i \in S \right\rangle$$

and

$$\left\langle \left(i, t^*, a_{i,1}^{(t^*)(1)}, \ldots, a_{i,l}^{(t^*)(1)}\right) \;\middle|\; i \in S \right\rangle ,$$

choose weights $\theta_j^{(t^*)}$, $1 \leq j \leq l$, for instance t^* and submit AO challenge series

$$\left\langle \left(i, t^*, \sum_{j=1}^{l} a_{i,j}^{(t^*)(0)} \theta_j^{(t^*)}\right) \;\middle|\; i \in S \right\rangle$$

and

$$\left\langle \left(i, t^*, \sum_{j=1}^{l} a_{i,j}^{(t^*)(1)} \theta_j^{(t^*)}\right) \;\middle|\; i \in S \right\rangle ,$$

to $\mathcal{C}$. The received response will be of the form

$$\left\langle (N+1)^{\sum_{j=1}^{l} a_{i,j}^{(t^*)(\beta)} \theta_j^{(t^*)}} H(t^*)^{\mathsf{sk}_{\mathsf{a},i}} \;\middle|\; i \in U \right\rangle ,$$

for an unknown $\beta \in \{0, 1\}$. Multiply series elements by $\tilde{\rho}_i^N$, $1 \leq i \leq n$, for randomly chosen $\tilde{\rho}_i \in \mathbb{Z}_N$ and return

$$\left\langle (N+1)^{\sum_{j=1}^{l} a_{i,j}^{(t^*)(\beta)} \theta_j^{(t^*)}} \tilde{\rho}_i^N H(t^*)^{\mathsf{sk}_{\mathsf{a},i}} \;\middle|\; i \in U \right\rangle$$

to $\mathcal{A}$.

Guess When $\mathcal{A}$ makes guess β', make the same guess β' to $\mathcal{C}$.

In the above construction, $\mathcal{C}$ follows the Joye-Libert scheme exactly, and to $\mathcal{A}$, $\mathcal{A}'$ follows our presented scheme exactly. Since $\mathcal{A}'$ runs in polynomial-time to security parameter when $\mathcal{A}$ does, and no non-negligible advantage adversary to $\mathcal{C}$ exists, we conclude that no non-negligible advantage adversary $\mathcal{A}$ exists. That is, there exists a negligible function η, such that

$$\mathsf{Adv}^{LCAO}(\mathcal{A}) \leq \eta(\kappa)$$

for security parameter κ. Lastly, the function H used by our scheme is treated as a random oracle in the Joye-Libert AO proof and will, therefore, prove our scheme secure in the random oracle model as well. □

List of Figures

List of Tables

List of Algorithms

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.