Online Self-Calibration for Robotic Systems

DISS. ETH NO. 21912

Online Self-Calibration for Robotic Systems

A thesis submitted to attain the degree of
Doctor of Sciences of ETH Zurich
(Dr. sc. ETH Zurich)

Presented by

Jérôme Maye
MSc en Informatique, Ecole Polytechnique Fédérale de Lausanne

Born November 13, 1981
Citizen of Chamoson (VS), Switzerland

Accepted on the recommendation of
Prof. Dr. Roland Y. Siegwart
Prof. Dr. Jan Peters
Dr. Paul Furgale

2014

Abstract

Every robotic system has some set of parameters—scale factors, sensor locations, link lengths, etc.—that are needed for state estimation, planning, and control. Despite best efforts during construction, some parameters will change over the lifetime of a robot due to normal wear and tear. In the best case, incorrect parameter values degrade performance. In the worst case, they cause critical safety issues.

In this thesis, we are interested in developing automated systems that are capable of robust and long-term deployment in the hands of non-experts, so the automatic identification and update of these parameter values is highly important.

We present a generic algorithm for self calibration of robotic systems that utilizes two key innovations. First, it uses an information-theoretic measure to automatically identify and store novel measurement sequences. This keeps the computation tractable by discarding redundant information and allows the system to build a sparse but complete calibration dataset from data collected at different times. Second, as the full observability of the calibration parameters may not be guaranteed for an arbitrary measurement sequence, the algorithm detects and locks unobservable directions in parameter space using a combination of rank-revealing QR and singular value decompositions of the Fisher information matrix. The result is an algorithm that listens to an incoming sensor stream, builds a minimal set of data for estimating the calibration parameters, and updates parameters as they become observable, leaving the others locked at their initial guess.

We demonstrate our online self-calibration algorithm on three applications, including a laser rangefinder on a differential-drive robot, odometry sensors of a consumer vehicle, and digital cameras with various lenses. Each application is fully described and validated through an extensive set of simulated and real-world experiments.

Résumé

Tout système robotique a un certain nombre de paramètres—facteurs d'échelle, positions des capteurs, longueurs de liaison, etc.—qui sont nécessaires pour l'estimation de son état, la planification de ses tâches et son contrôle. Malgré tous les efforts entrepris lors de la construction, certains paramètres ont tendance à changer au cours de la durée de vie d'un robot en raison de l'usure normale. Dans le meilleur des cas, des valeurs incorrectes pour les paramètres dégradent les performances. Dans le pire des cas, elles peuvent provoquer des problèmes de sécurité critiques.

Dans cette thèse, nous nous intéressons au développement de systèmes automatisés qui peuvent être déployés de manière robuste et sur le long terme par des personnes non expérimentées. Dans cette optique, l'identification automatique et la mise à jour des ces valeurs relèvent d'une importance capitale.

Nous présentons un algorithme générique pour la calibration automatique de systèmes robotiques axé sur deux innovations majeures. Tout d'abord, il utilise une mesure basée sur la théorie de l'information, qui permet d'identifier et de stocker automatiquement de nouvelles séquences de données. Tout en réduisant la complexité computationelle en éliminant les données redondantes, cela permet de construire un jeu de données minimal pour la calibration à partir de données acquises à des moments différents. Deuxièmement, comme l'observabilité complète des paramètres de calibration ne peut être garantie pour une séquence arbitraire de données, l'algorithme détecte et bloque les directions non observables dans l'espace des paramètres en utilisant une combinaison de décomposition QR révèlant le rang et de décomposition en valeurs singulières de la matrice d'information de Fisher. Cela résulte en un algorithme qui traite un flux de données sensorielles, construit un jeu de données minimal pour estimer les paramètres de calibration, et met à jour ces paramètres au fur et à mesure qu'ils deviennent observables, tout en laissant les autres verrouillés à leur estimation initiale.

Nous démontrons la pertinence de notre algorithme incrémental de calibration automatique à l'aide de trois applications, comprenant un télémètre laser monté sur un robot de type differentiel, des capteurs d'odométrie d'une voiture de tourisme, et des appareils photo numériques avec différentes lentilles. Chaque application est présentée de manière détaillée et est validée par un ensemble de simulations et d'expériences avec des données réelles.

Contents

1 Introduction

The shortest path between two truths in the real domain passes through the complex domain.

Jacques Hadamard (1865-1963)

1.1 Motivation and Objectives

Every robotic system has some set of parameters—scale factors, sensor locations, link lengths, etc.—that are needed for state estimation, planning, and control. Despite best efforts during construction, some parameters will change over the lifetime of a robot due to normal wear and tear. In the best case, incorrect parameter values degrade performance. In the worst case, they cause critical safety issues.

We are interested in developing automated systems that are capable of robust and long-term deployment in the hands of non-experts, so the automatic identification and update of these parameter values is highly important.

As an example, consider a camera-based collision avoidance system intended for deployment in a consumer automobile. For the vehicle to be able to avoid collisions, the pose of each camera with respect to the vehicle coordinate system must be known precisely so that obstacle positions can be transformed from camera coordinates into vehicle coordinates. However, a consumer vehicle will have no access to special calibration hardware or expert data analysis. In such a scenario, the vehicle must be capable of self-supervised recalibration. This problem is inherently difficult for a number of reasons that we will briefly discuss here.

1) Parameters change over time—although the vehicle may be factory calibrated, the transformations can change slowly over time due to vibration, thermal expansion, loose parts, or any number of other common problems that follow from normal usage.

2) Parameters must be inferred from the data—as the cameras may be installed in different places on the vehicle, their pose cannot be measured directly. Instead, the transformations must be inferred from the data produced by the full system. However, this is only possible if the motion of the vehicle renders the parameters *observable*[1].

3) Normal operation may result in unobservable directions in parameter space—unfortunately, for this and many other practical problems normal operation may not render all directions in parameters space observable. In this example, when two cameras do not share an overlapping field of view, planar motion renders the calibration problem degenerate[2]; the transformation between cameras only becomes observable under general 3D motion.

[1] Informally, observability means that the parameters can be inferred from some local batch of measurement data [Jaz70].

[2] See [LRAAD10] for a handy table of degenerate cases or [KC06] for a more theoretical analysis.

4) Unobservable directions in parameter space may appear observable in the presence of noise—even if our hypothetical car is piloted only on a plane (a degenerate case), noise in the measurements can make unobservable parameters appear observable. We call these parameters *numerically unobservable* to mirror the related concept of numerically rank-deficient matrices.

Existing algorithms for self calibration generally handle issues (1) and (2). Issue (3) is dealt with by designing experiments that guarantee all parameters become observable. To the best of our knowledge, no published self-calibration algorithm is able to cope with issue (4). Therefore, unless we plan to require all vehicle owners to outfit their parking places with calibration patterns or regularly drive off-road, new advances for online system calibration are required.

The V-Charge project[3] offers a motivating example that illustrates the above points. The main objective of this European project is to develop an automated driving system that solely relies on low-cost sensors such as cameras. As shown in Fig. 1.1, the platform is a regular Volkswagen Golf equipped with four cameras having non-overlapping field of view, that are smoothly integrated into the wing mirrors and Volkswagen signs. Due to their location on the vehicle, the relative transformations between the cameras and the vehicle coordinate system are varying over time, *e.g.*, whenever the side doors are opened or closed. In Fig. 1.2, we have plotted the evolution over time of the relative translation vectors between the vehicle coordinate system and the four cameras. These values result from several calibration sessions executed over a period of one year. We observe from these plots that the frequent variations of the calibration parameters call for an efficient and simple calibration method. For a large deployment into consumer vehicles, the calibration algorithm should furthermore be entirely automated and continuously run in the background without any human intervention. It would indeed be an economic nonsense to regularly get these cars calibrated by professionals in a dedicated area.

To conclude on the V-Charge example, it is worth depicting the effect of an incorrect sensor calibration on the vehicle operation. Fig. 1.3 shows the top-view image of a parking place reconstructed from the four vehicle cameras. In this situation, inaccurate calibration parameters can cause the car to be badly parked or to hit an obstacle in the best case. During autonomous driving on typical city streets, this might however lead to more dramatic consequences involving human casualties. Therefore, accurate sensor calibration is of paramount importance and constitutes the building block of any well-functioning robotic system.

In this thesis, we propose an algorithm to deal with *all* of the aforementioned difficulties. Our approach exploits the algebraic links between the Gauss-Newton algorithm, the Fisher information matrix, and nonlinear observability analysis to automatically detect directions in parameter space that are numerically unobservable and avoid updating our parameters in these directions; at any given time, directions in the parameter space that are observable will be updated based on the latest information, while unobservable directions will remain at their initial guess. Novel sets of measurements are detected using an information gain test, and added to a working set that is used to estimate the parameters. The result is an algorithm that listens to an incoming stream of data, automatically accumulating a batch of data that may be used to calibrate a full robot system. The only requirements are that (i) the parameters are the-

[3] http://www.v-charge.eu/

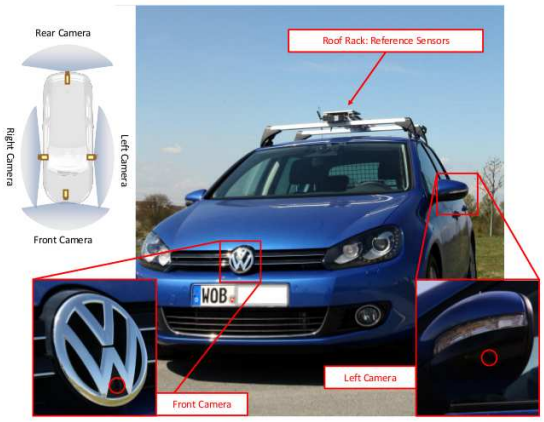
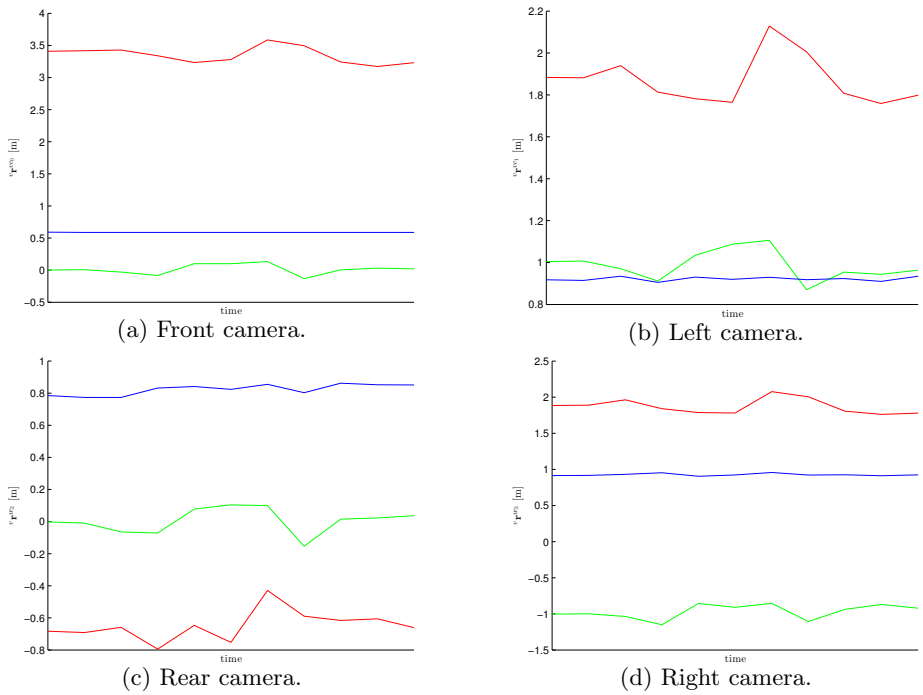


Figure 1.1: The V-Charge platform is a regular Volkswagen Golf equipped with four cameras having non-overlapping field of view (image courtesy of Peter Muehlfellner, 2014). Two cameras are mounted in the wing mirrors and two cameras in the front and rear Volkswagen signs. The reference sensor depicted here is a combination of Global Navigation Satellite System (GNSS) and Inertial Navigation System (INS), which provides ground truth to the camera-based navigation algorithms.



(a) Front camera.

(b) Left camera.

(c) Rear camera.

(d) Right camera.

Figure 1.2: Evolution of the the translation vectors between the vehicle coordinate system and the four cameras of the V-Charge platform over a period of one year. For each plot, the red curve represents the first vector component, the green curve the second vector component, and the blue curve the third vector component.

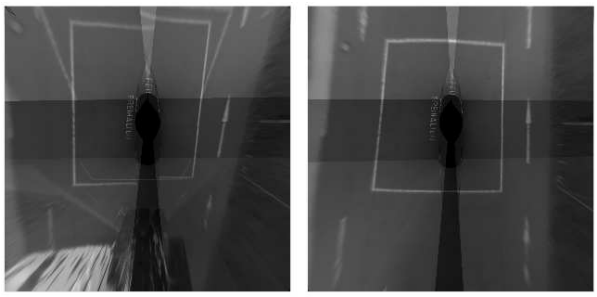


Figure 1.3: Top-view image of a parking place reconstructed from the four cameras of the V-Charge vehicle (image courtesy of Lionel Heng, ETH Zurich, 2014). While the left image results from incorrect calibration parameters, the right image accurately represents the parking place and allows for safe navigation within the parking lot.

oretically observable given some ideal set of data, (ii) it is possible to implement a batch Gauss-Newton estimator for the system state and calibration parameters based on a set of measurements, and (iii) we have some reasonable initial guess for the calibration parameters (*e.g.* from factory calibration or manual measurements).

1.2 Overview of Related Work

This section is dedicated to the literature review of the most prominent related works in the various domains spanned by our contributions. On the one hand, we shall outline the advantages and disadvantages of each method, and on the other hand put them in contrast to our approach. Our thesis being primarily targeted at a roboticist audience, we shall start with the general topic of sensor calibration and present the relevant publications of the past decade. Departing from the main application, we shall then delve into regularization approaches for linear and nonlinear regression problems. Finally, we will conclude the section with contributions related to our online algorithm.

1.2.1 Sensor Calibration

The problem of sensor calibration has been a recurring one in the history of robotics and computer vision. Thereby, it has been addressed using a variety of sensor setups and algorithms. A calibration process may involve recovering *intrinsic*, *e.g.*, focal length for a camera, and *extrinsic* parameters, *i.e.*, the rigid transformation between the sensor's coordinate system and a reference coordinate system. For the former, one could devise a naive approach where the parameters are accurately determined during the manufacturing process. Similarly, the transformation could be retrieved by means of some measuring instrument. However, the disadvantages of such purely engineered methods are manifold. Apart from their impracticality, it can be nearly impossible to reach a satisfying level of accuracy and thus hinder the proper use of the sensor in a robotic system. Furthermore, external factors such as temperature variations or mechanical shocks may seriously bias a factory calibration. Therefore, much

efforts have been dedicated over the years to develop algorithms for calibration of systems in the field.

At a first level, sensor calibration methods can be classified into online or offline algorithms. While the latter are more popular in the computer vision community, the former are widespread in robotic applications. Online methods usually augment the state vector of a Bayes filter with calibration parameters and offline approaches rely on the minimization of a cost function. At a second level, calibration algorithms can be distinguished in how they address observability or singularity issues. While some publications present an informal or formal offline analysis, some others silently assume that the dataset contains enough information to calibrate the sensor parameters. Finally, the complexity of the setup and the amount of supervision are substantial criteria to assess the practicality of a calibration method.

Due to its simplicity, the use of a planar calibration pattern such as a checkerboard (*e.g.* in Fig. 1.4), coupled with nonlinear regression, has become the most popular method in computer vision during the last decade for calibrating intrinsic camera parameters [Zha99, SM99]. Given a set of point correspondences between the model planes and their images, the algorithm attempts to minimize a nonlinear cost function consisting in the sum of squared reprojection errors. While [Zha99] remains rather vague about degenerate configurations, a compendium of singularities is proposed in [SM99]. The planar calibration pattern is also used in [ZP04] to estimate the relative pose between a camera and a laser rangefinder, with no particular attention to singularities in the camera views. While being relatively efficient and inexpensive, this procedure still requires expert knowledge to reach a good level of accuracy. Even though point correspondences can be automatically identified by corner detectors [HS88, RSS08], it is still up to the user to execute proper motion of the calibration pattern in front of the camera.

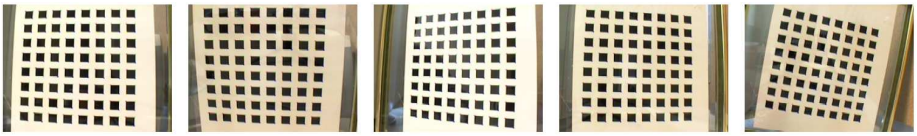


Figure 1.4: Five images of the calibration pattern used in [Zha99]. Given the known 3D geometry of the pattern, corners extracted in each view can be associated to world points and the sum of squared reprojection errors is minimized.

A key limitation of the plane-based calibration technique for intrinsic camera parameters is that it remains essentially offline. Whenever a change of calibration occurs, *e.g.*, an adjustment of the focal length, the vision task has to be interrupted for recalibration. In order to allow for online operation, Maybank and Faugeras have pioneered in [MF92, FLM92] a method for self-calibration that does not require any pattern with known 3D geometry. Given a couple of images from the same scene under different viewpoints and point correspondences as shown in Fig. 1.5, epipolar transformations are computed and a system of equations is solved to yield the camera intrinsic parameters. Due to the lack of probabilistic model, the method is inherently sensitive to noise. Furthermore, since a proper calibration cannot be guaranteed under general camera motion, the algorithm is not mature for large-scale deployment.

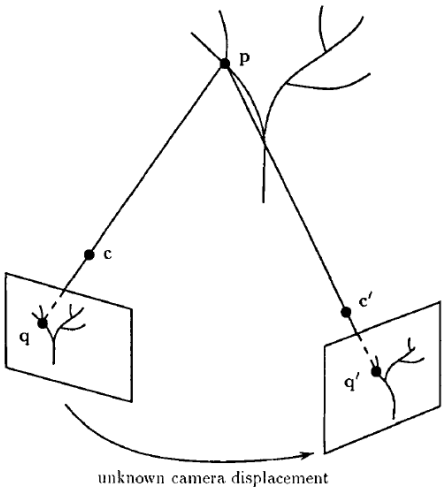
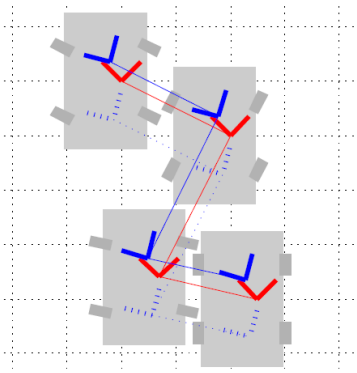


Figure 1.5: Example of point correspondence for the method in [MF92]. From a series of images of a scene under different viewpoints, epipolar transformations established from the point correspondences are used to construct a system of quadratic equations, whose solution provides the camera intrinsic parameters.

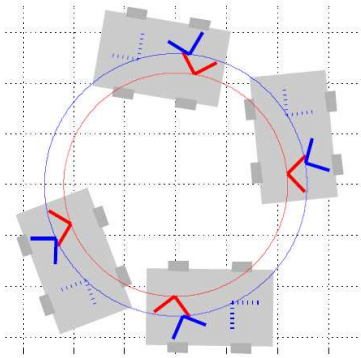
In the context of mobile robotics, several authors have included the calibration problem in a state estimation framework, either with filtering [MSS06, MR08, GS10, KS11] or nonlinear least-squares [CMO08, KGB11, HLP13] techniques. Filtering techniques based on the Kalman filter are appealing due to their inherently online nature. However, in case of nonlinear systems, least-squares techniques based on iterative optimization are usually superior in terms of accuracy. In [MSS06], Martinelli *et al.* perform a nonlinear observability analysis using Lie derivatives and overcome the singularities by choosing an alternative parameterization for the robot's state. An identical method for observability analysis has also been employed in [MR08, KS11].

More recently, Brookshire and Teller [BT11, BT12] have carried out a formal observability analysis in order to identify degenerate paths of the calibration dataset as depicted in Fig. 1.6. In contrast to the aforementioned methods, they inspect the rank of the Fisher information matrix to determine the observability of the calibration parameters. However, their approach still expects that the robot travels along non-degenerate paths during the calibration run. A standard nonlinear least-squares method is subsequently used for the estimation.

Another class of methods seeks to maximize the concordance of the sensor data with the environment [UHS07, LT10] or a quality measure [SHN10, MHN12, SHN12]. While Underwood *et al.* [UHS07] use a vertical pole with retro reflective tape, Levinson and Thrun [LT10] propose an unsupervised approach relying on the assumption that neighboring points from laser beams lie on a contiguous surface as shown in Fig. 1.7. The method presented in [SHN10, MHN12, SHN12] estimates relative poses of laser rangefinders by minimizing the entropy of an underlying generative model for the location of laser points in space, hence maximizing the sharpness of the distribution and the quality of the resulting point cloud as displayed in Fig. 1.8.



(a) Translational motion.



(b) Rotational motion.

Figure 1.6: Examples of degenerate paths discovered by the algebraic analysis of the Fisher information matrix in [BT11]. The calibration problem becomes singular for purely translational (Fig. 1.6a) or rotational (Fig. 1.6b) motion.

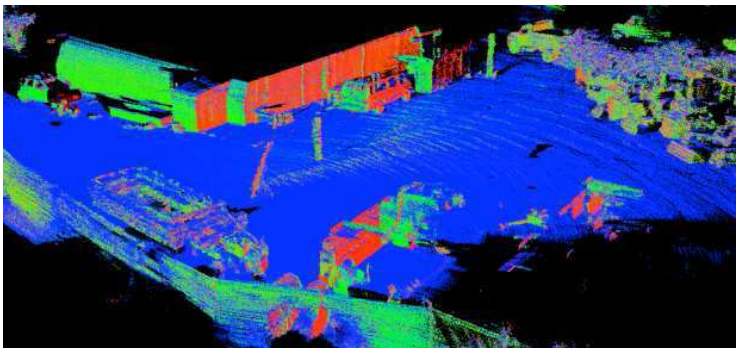


Figure 1.7: Example of point cloud from [LT10] colored by surface normal. The algorithm relies on the assumption that neighboring laser beams lie on a contiguous surface.

(a) Uncalibrated laser.

(b) Calibrated laser.

Figure 1.8: Examples of point clouds acquired by rotating laser rangefinders in [SHN12]. Starting from an uncalibrated setup (Fig. 1.8a), the algorithm optimizes a cost function that improves the crispness of the resulting point cloud (Fig. 1.8b).

Closely related to our approach, Menq *et al.* [MBL89] have also adopted matrix factorization to identify observable and unobservable subspaces in the context of calibration of industrial robots. However, the authors do not explicitly deal with the notion of numerical rank deficiency, which, in our opinion, is essential to the proper operation of a calibration algorithm.

To the best of our knowledge, little research has been devoted to account for degenerate cases frequently occurring during calibration runs. Some authors, notably in the robotic community, have proposed an offline observability analysis to either state that the system is observable or to identify potential singularities. As will be demonstrated in this thesis, traditional observability analysis is helpless when it comes to noisy input data on real physical systems. It is indeed purely algebraic, performed a priori, and hence omits the numerical difficulties associated with sensory data. In general, apart from [MBL89], the authors assume that their estimation framework is fed with well-behaved data. In contrast, our approach relaxes this assumption and is therefore suitable in the hands of non-expert users for online and long-term operations on various platforms and sensors.

1.2.2 Regularization in Least-Squares Problems

Mostly absent from the sensor calibration algorithms, regularization techniques for linear and nonlinear regression are central to many estimation problems in numerical analysis. In that regard, part of this thesis takes inspiration on the inverse problem theory and we will largely elaborate on that topic in the following chapters.

In numerical linear algebra, matrix analysis is the counterpart of observability analysis. In this field, real-world data often lead to ill-conditioned or ill-posed problems. To still provide usable solutions to these problems, a common method is to employ regularization [Han98]. Regularization can be achieved by adding a full-rank diagonal matrix to the data matrix (*e.g.*, Tikhonov regularization), by a low-rank approximation of the data matrix (*e.g.*, truncated SVD), or by subset selection of the most linearly independent columns of the data matrix. Most of the theory has been articulated for linear systems, but has natural extensions to nonlinear systems [Cha09, IKP11]. Our thesis does not pretend to compete in this well-established field of research and we do not propose any novel theoretical insights in that direction. However, we hope to bring some practical contributions with a complete methodology for solving real-world problems involving large and noisy datasets.

A substantial body of literature on regularization exists in the control-theory community [FKY81, SEF85, CS00, EM06, EM09]. In these papers, the goal is to devise an adaptive controller with a recursive least-squares estimator and a forgetting factor discarding old information. If the new data entering the filter does not contain enough information about the state being estimated, *i.e.*, it does not provide enough *excitation*, the information matrix becomes singular. The principal reason for this so-called wind-up or blow-up effect is that the old data is discarded regardless of whether it can be replaced by new data. In [FKY81], the authors propose a variable forgetting factor strategy that keeps the information content of the filter constant over time. In [CS00], Cao and Schwartz use a decomposition of the information matrix to determine the optimal direction of the forgetting factor along the exciting direction. This

idea bears a strong resemblance to our approach. While in their algorithm the forgetting occurs only in the direction of excitation, we only perform updates of the parameters in the observable direction. Similarly, the orthogonal direction of excitation is left unchanged to avoid singularities in the information matrix.

1.2.3 Measurement Selection and Sparsification

During the past decades, information theory has largely paved his way into statistical estimation problems, particularly in the robotic and computer vision community. In [Dav05, KS12b], the authors employ a comparable strategy to what will be presented in this thesis, *i.e.*, discarding measurements based on their utility. While Davison proposes a closed-form expression for the mutual information between the current state vector and a measurement vector, Kretzschmar and Stachniss approximate it using an occupancy grid map. For our calibration algorithm, we claim that the mutual information is not a suitable measure for estimating the utility of incoming measurements. The mutual information formula involves an expectation over a joint probability density function, *i.e.*, it can be computed regardless of the actual observed values of the random variables. For the least-squares problems we are tackling, the local observability of the parameters critically depends on the observed data and we are thus unable to evaluate the utility of an incoming batch of data a priori. As in [SGB05], our algorithm uses the entropy difference to calculate an information gain. Nevertheless, Stachniss *et al.* also compute an expected information gain in this situation, since it has to guide the robot for selecting a particular action based on its a priori utility for the reduction of the uncertainty of the filter.

The problem of online learning has also been tackled in the context of kernel-based classification or regression. As in our approach, these methods are inherently offline since they require the entire training dataset for performing prediction and the computation rapidly becomes intractable as the number of training data grows. To circumvent this issue, Engel *et al.* propose in [EMM04] a recursive algorithm, coined constructive online *sparsification*, that inserts an incoming measurement into a *dictionary* based on its linear independence to the already incorporated measurements. The authors have further shown that their sparsification method can be interpreted as an approximate principal component analysis (PCA), *i.e.*, a dimensionality reduction of the training data. Although bounded, the dictionary can grow very large for a given threshold on the linear independence. To allow for real-time operation, Nguyen-Tuang and Peters have extended the approach in [NTP11] to constrain the size of the dictionary to a fixed *budget*. When the dictionary reaches the allocated budget, measurements are removed based on their linear dependence to the others and a forgetting factor that penalizes old data. While the online sparsification algorithm directly acts on the input data prior to estimation, our method is specifically tailored to assess the utility of the measurements based on the reduction of uncertainty of the estimated quantity. The sparsification method is furthermore tightly linked to the supervised learning paradigm with input/output training data points of fixed dimensionality. In contrast, we only receive output data points, which might be of varying dimensionality in a multi-sensor setup.

1.3 Contributions

This section is intended to outline the main contributions of this thesis. The detailed explanations will be reported within the relevant parts of the document and we shall here briefly summarize the major points.

The central contribution of this work pertains to a new perspective on observability analysis for calibration problems. While the vast majority of existing approaches adopt an algebraic view on observability, we have pioneered an online and numerical method that continuously quantifies the observability of the calibration parameters based on incoming batches of data. Although being relevant as an a priori measure for an observation model, traditional observability analysis does not help in avoiding singularities during the calibration runs. Furthermore, noisy input data introduce numerical difficulties that cannot be handled in an offline phase. Instead of a binary statement, our algorithm provides a measure for the degree of observability of each dimension of the parameter space. In that sense, our method is suitable for unsupervised calibration as it does not require any particular action of the user; parameters will solely be updated when the corresponding direction becomes observable, leaving the unobservable subspace intact.

Relying on nonlinear optimization techniques, our algorithm is inherently offline, *i.e.*, it requires a batch of data to minimize a cost function. Unlike state-of-the-art approaches, we overcome this limitation with an information-theoretic scheme that selects batches of data based on their utility for the calibration. Due to the consistency of the underlying estimator, the calibration dataset will not grow indefinitely and our algorithm is hence suitable for online and long-term operations. Therefore, we might classify our method in the *online batch* or *sequential batch* category.

A third contribution of this work concerns the detailed character of the algorithmic and experimental parts of the thesis, which shall encourage the reader to adopt our method for his particular application. Every piece of MATLAB and C++ code used for the experimental evaluation, along with the datasets, is furthermore freely available on demand.

Lastly, we have taken the opportunity in Chap. 2 to summarize part of the knowledge gathered over the years along with the relevant references. While writing this chapter, the underlying objective was to transmit these concepts to a large audience without preliminary knowledge on the topics and we do hope to stimulate the curiosity of the reader.

In a more compact representation, the core contributions of this thesis can be summarized as follows:

- Novel perspective on observability analysis for calibration problems
- Online algorithm based on an information-theoretic measure
- Validation of the framework on three challenging applications

1.4 Thesis Outline

The thesis starts by establishing a solid theoretical ground in Chap. 2 that will guide the reader through the thesis. On the one hand, this chapter shall lay down the mathematical notation and on the other hand introduces the major concepts in statistical estimation and linear algebra. An experienced reader familiar with these concepts might skip some of the sections and refer back to them during the course of the following chapters.

In Chap. 3, we present the central contributions of the thesis. Starting from a formal description of the theoretical parts of our method, we conclude with a practical section targeted at a straightforward implementation.

In Chap. 4, we suggest and demonstrate three practical applications for our online self-calibration algorithm. Every application is validated through a set of experiments and accompanied with a complete description which shall facilitate the deployment of our method to other calibration problems.

Finally, Chap. 5 concludes the work by summarizing the main contributions and stating some of the limitations. Undoubtedly, our work opens the door to diverse improvements and further research that shall be outlined in this chapter.

To ease navigation, we have summarized the thesis outline in Fig. 1.9. Due to the numerous hyperlinks facilitating cross-referencing through the thesis, we recommend the reading of this document in its electronic version. Furthermore, most of the figures are embedded in vectorized format to allow arbitrary scaling for exploring the details. The important keywords are italicized and listed in the index at the end of the document.

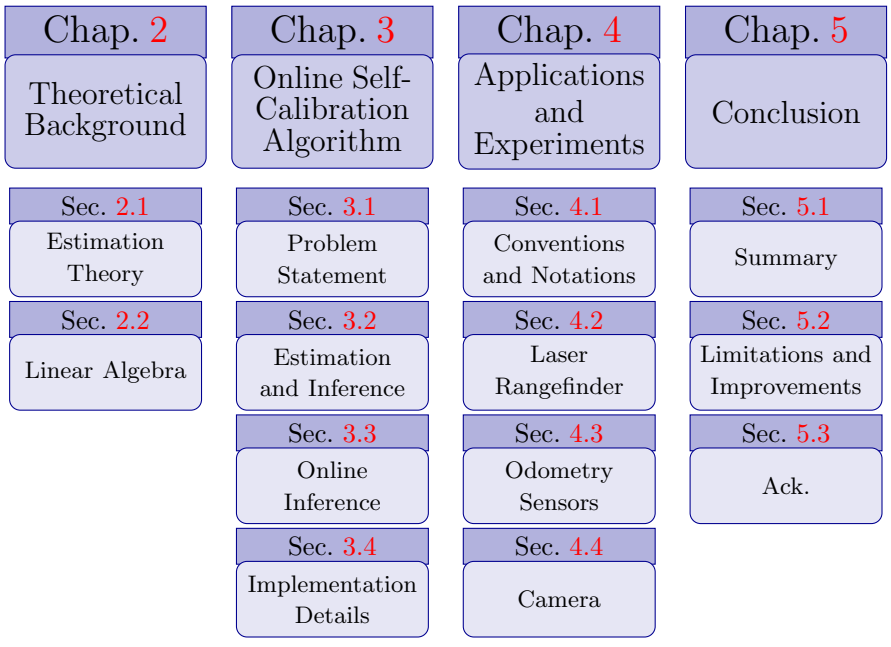


Figure 1.9: Visual outline of the thesis.

2 Theoretical Background

If I have seen further it is by standing on the shoulders of giants.

Sir Isaac Newton (1642-1727)

The theoretical part of this thesis is inspired by the work of the greatest minds of science whom we want to acknowledge in this chapter. The way we conceive science nowadays finds its roots in the seventeenth century with the work of Francis Bacon on the scientific method and has been largely influenced by the seminal books of Ronald Fisher in the twentieth century [Fis25, Fis35]. After the model design phase, experimentation is at the heart of the scientific method. Since the output of an experiment is often subject to random variations, a tremendous body of work on statistical estimation has emerged to provide conclusive outcomes. Along with statistical theory, linear algebra theory has brought crucial methods to estimate complex multivariate models. In particular, with the advent of modern computers, numerical analysts have proposed sound and detailed algorithms for dealing with estimation problems.

This chapter is devoted to the main principles of statistical theory and linear algebra. It is rather uncommon to present them together, but we feel that both fields can benefit from each other. Furthermore, our thesis borrow techniques from both disciplines. This chapter ought not to be considered as a textbook, but rather as a cheat sheet for the next chapters. We will essentially highlight the key theoretical elements that justify our approach and refer the reader to further publications. Although an experienced reader might skip this chapter, it remains valuable for a deep understanding of the core of our approach. Moreover, the selection and the interpretation of the scientific literature are intrinsically subjective and reflect the views of the author.

2.1 Estimation Theory

Estimation theory aims at inferring some quantities of interest or parameters based on empirical measurements containing random components. All the involved quantities are linked through a mathematical model describing the underlying physical setting. For instance, exoplanets can only be observed indirectly through their radio emission measured by radio telescopes which are inherently subject to electrical noise. History of experimental sciences has indeed shown that estimation theory is a key element to any experiment involving a measurement process, in various fields such as astrology, biology, or physics. This mainly stems from the employed sensing technologies, generally affected by systematic and random errors. Estimation theory is a vast field of research spread over centuries comprising various aspects and viewpoints that we intend to present in this section.

2.1.1 Probability Theory

Probability theory is a theoretical branch of mathematics that studies random phenomena based on a set of axioms. The concept of probability has a long history that finds its roots in the sixteenth century in the game of chance and has been formalized and unified by Kolmogorov [Kol50] in the twentieth century. This latter is considered the precursor of modern probability theory. The rest of this section will briefly outline the main concepts of this theory that provides the building block for reasoning under uncertainty. We will essentially adopt an algebraic view and therefore avoid the formalism of the set theory. In a similar vein, we will elude the philosophical debate on the interpretation of probabilities and merely consider probability theory as a tool for assessing the uncertainty or variability of some quantities of interest. For a more in-depth treatment, we refer the reader to [Kol50] or any modern textbook on the matter.

Probability Density Function

A *random variable* is a mathematical variable that can take values from a potentially infinite set of discrete or continuous values, known as the *sample space*, and is associated with a *probability distribution* determining a rule for the selection of those values. A particular outcome or realization of a random variable is a *random variate*. Formally, we denote a random variable as X, its probability distribution as f_X, its sample space as Ω_X, and a random variate as x. When a random variable takes values from a continuous set, it is known as a *continuous random variable* and its associated probability distribution is usually specified by a *probability density function*. Otherwise, it is called a *discrete random variable* and its probability distribution is commonly represented by a *probability mass function*. In this thesis, we will primarily manipulate continuous random variables and hence omit the specific derivations for discrete random variables.

A probability density function is a mathematical function that satisfies the following criteria

$$\begin{aligned} f_X(x) &\geq 0, \forall x \in \Omega_X, \\ \int_{\Omega_X} f_X(x)\,\mathrm{d}x &= 1. \end{aligned} \tag{2.1}$$

Amongst the various representations for a probability distribution, it is worth defining the *cumulative distribution function*, or simply the *distribution function*, as

$$F_X(x) = \int_{\min \Omega_X}^{x} f_X(x)\,\mathrm{d}x, \tag{2.2}$$

and its inverse F_X^{-1}, also known as the *quantile function*. Formally, a probability density function is defined in term of a distribution function.

In order to make statements about the values that a random variable might take, we need to introduce the notion of *probability*. A probability is a numerical measure in the range $[0, 1]$ that determines how likely a random variable will take specific values from the sample space. For continuous random variables,

the probability that a random variable takes exactly a specific value is zero by definition. In that case, we can express the probability that a random variable falls into an interval $[x_a, x_b]$ as

$$p(x_a \leq X \leq x_b) \quad = \quad \int_{x_a}^{x_b} f_X(x)\, \mathrm{d}x, \tag{2.3}$$

or that a random variable is less than a value x as

$$p(X \leq x) \quad = \quad F_X(x). \tag{2.4}$$

From Equ. 2.4, we notice that the cumulative distribution function offers a direct link to probabilities and that Equ. 2.3 can be formulated differently as

$$p(x_a < X \leq x_b) \quad = \quad F_X(x_b) - F_X(x_a). \tag{2.5}$$

When a random variable is associated with a specific probability density function, we will use the notation

$$X \quad \sim \quad f_X. \tag{2.6}$$

Joint, Marginal, and Conditional Probability Density Function

For a set of n random variables $\{X_1, \ldots, X_n\}$, we can define a *joint probability density function*, which we denote as $f_{X_1,\ldots,X_n}$. The random variables can further be assembled into a *multivariate random variable* or *random vector*, which we denote in vectorial form as $\mathbf{X} = (X_1, \ldots, X_n)^T$. A *multivariate random variate* will be written as $\mathbf{x} = (x_1, \ldots, x_n)^T$. Henceforth, we shall use the set and boldface notations interchangeably depending on the context and limit our discussion to the more general multivariate setting. The above equations and definitions, derived for *univariate random variables* and *univariate random variates*, can be extended to the multivariate case. In particular, the criteria for density functions in Equ. 2.1 are transformed into

$$\begin{aligned} f_{\mathbf{X}}(\mathbf{x}) \quad &\geq \quad 0, \forall \mathbf{x} \in \Omega_{\mathbf{X}}, \\ \int_{\Omega_{\mathbf{X}}} f_{\mathbf{X}}(\mathbf{x})\, \mathrm{d}\mathbf{x} \quad &= \quad 1. \end{aligned} \tag{2.7}$$

In Equ. 2.7, we have adopted a simplified notation for the multiple integral on the n-dimensional space $\Omega_{\mathbf{X}}$.

In terms of probabilities on this space, we can generalize Equ. 2.4 to

$$p(X_1 \leq x_1, \ldots, X_n \leq x_n) \quad = \quad F_{\mathbf{X}}(\mathbf{x}), \tag{2.8}$$

where $F_{\mathbf{X}}$ is a *joint cumulative distribution function*. We can also extend Equ. 2.3 to higher dimensions and express the probability that the random variables will take values into an n-dimensional domain $D \subset \Omega_{\mathbf{X}}$ as

$$p(\mathbf{X} \in D) \quad = \quad \int_D f_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}\mathbf{x} \tag{2.9}$$

For all X_i in the set, we can obtain a *marginal density function* f_{X_i} by integrating over the other variables, *i.e.*,

$$f_{X_i} \quad = \quad \int \cdots \int f_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}x_1 \ldots \, \mathrm{d}x_{i-1} \, \mathrm{d}x_{i+1} \ldots \, \mathrm{d}x_n. \tag{2.10}$$

In Equ. 2.10, we have omitted the domain of the multiple integral, which is defined as the $(n-1)$-dimensional subspace of $\Omega_{\mathbf{X}}$ with the variable X_i dropped out. When it is clear from the context, we will systematically omit the domain below the integral sign.

The random variables in a set are *independent* if and only if

$$f_{\mathbf{X}}(\mathbf{x}) \quad = \quad \prod_{i=1}^{n} f_{X_i}(x_i). \tag{2.11}$$

Intuitively, if random variables are independent, the realization of one does not affect the distribution of the others. This concept is crucial to the *factorization* of complex multivariate random variables.

If we now introduce an additional random variable $\mathbf{Z}$, we can formulate the *conditional probability density function* of $\mathbf{X}$ given an occurrence $\mathbf{z}$ of $\mathbf{Z}$ as

$$f_{\mathbf{X}}(\mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \quad = \quad \frac{f_{\mathbf{X},\mathbf{Z}}(\mathbf{x}, \mathbf{z})}{f_{\mathbf{Z}}(\mathbf{z})}, \tag{2.12}$$

where $f_{\mathbf{X},\mathbf{Z}}$ is the joint probability density function of $\mathbf{X}$ and $\mathbf{Z}$, and $f_{\mathbf{Z}}$ the marginal probability density function of $\mathbf{Z}$. The independence of random variables can be expressed in terms of conditional densities, *i.e.*, $\mathbf{X}$ and $\mathbf{Z}$ are independent if and only if, for all $\mathbf{z}$ in $\Omega_{\mathbf{Z}}$, we have

$$f_{\mathbf{X}}(\mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \quad = \quad f_{\mathbf{X}}(\mathbf{x}). \tag{2.13}$$

Finally, the random variables $X_1, \ldots, X_n$ are *conditionally independent* given the occurrence $\mathbf{z}$ of $\mathbf{Z}$ if

$$f_{\mathbf{X}}(\mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \quad = \quad \prod_{i=1}^{n} f_{X_i}(x_i \mid \mathbf{Z} = \mathbf{z}). \tag{2.14}$$

As for Equ. 2.11 and Equ. 2.13, conditional independence plays a significant role in the simplification of complex probabilistic models.

Characterization of Probability Density Functions

When working with probability density functions, there are various properties that characterize their location and shape in space. In particular, the concepts of *moment* and *central moment* provide several important quantitative measures. The n-th moment of a probability density function is defined as

$$\mathbb{E}\left[\mathbf{X}^n\right] = \int_{\Omega_{\mathbf{X}}} \mathbf{x}^n f_{\mathbf{X}}(\mathbf{x}^n)\, \mathrm{d}\mathbf{x}. \tag{2.15}$$

A moment is referred to as the *expected value* of $\mathbf{X}^n$. In the case of $n = 1$, it corresponds to the *mean* of the density function, *i.e.*, its center of mass or location. A central moment of a probability density function corresponds to a moment around its mean, *i.e.*, $\mathbb{E}\left[(\mathbf{X} - \mathbb{E}\left[\mathbf{X}\right])^n\right]$. In the case of $n = 2$, we retrieve a *covariance matrix*, defined as

$$\mathrm{Var}\left[\mathbf{X}\right] = \mathbb{E}\left[(\mathbf{X} - \mathbb{E}\left[\mathbf{X}\right])(\mathbf{X} - \mathbb{E}\left[\mathbf{X}\right])^T\right]. \tag{2.16}$$

The diagonal elements of the covariance matrix correspond to *variances* and are denoted $\mathrm{Var}\left[X_i\right] = \mathrm{Cov}\left[X_i, X_i\right]$. The variance characterizes the spread of the probability density function around its mean. The square root of the variance is known as the *standard deviation*. The off-diagonal elements measure how variations in one of the variable influence another variable and are denoted $\mathrm{Cov}\left[X_i, X_j\right] = \mathrm{Cov}\left[X_j, X_i\right]$, or the covariance between X_i and X_j. The geometric interpretation of the covariance is the orientation of the density function in space. For some applications, it can be more intuitive to use the concept of *correlation*, which is the normalized version of the covariance.

A probability density function can further be characterized by its *mode*. This corresponds to its maximal value and is denoted as

$$\mathrm{Mode}\left[\mathbf{X}\right] = \underset{\mathbf{x}}{\mathrm{argmax}}\, f_{\mathbf{X}}(\mathbf{x}). \tag{2.17}$$

In the case where the mode is unique, the distribution is *unimodal*. Otherwise, it is *multimodal*. Intuitively, the mode defines the most likely value to be assigned to a random variable.

A *quantile* is the value $\mathbf{x}$ that divides the probability mass proportionally to a given probability p. Formally, this can be expressed as the random variate $\mathbf{x}$ that satisfies

$$F_{\mathbf{X}}(\mathbf{x}) = p. \tag{2.18}$$

If there exists an analytical form for the inverse cumulative distribution function, the quantile can be expressed as

$$\mathbf{x} = F_{\mathbf{X}}^{-1}(p). \tag{2.19}$$

Some quantiles are widely used in the literature and are given proper names. In particular, we can mention the *median* ($p = 0.5$), the first quartile ($p = 0.25$), and the third quartile ($p = 0.75$). Depending on the shape of the density, the median can be a more relevant measure for the location.

2.1.2 Information Theory

Information theory originates from the work of Shannon [Sha48] on the transmission of coded data over a noisy channel. Over the years, it has been the cornerstone to many technical developments such as compression algorithms or the compact disc. The main concept of information theory is the *information*, which can be literally defined as a piece of knowledge communicated or received about a particular fact.

Discrete Random Variables

In Shannon's work, information is a purely abstract mathematical concept, *i.e.*, its semantic content is not relevant. Information is represented by a message x from an alphabet Ω_X with a known probability distribution f_X. The *information content* of a discrete random variable X with a realization x is defined as

$$I(x) \quad = \quad -\log_b p(X = x), \tag{2.20}$$

where the basis of the logarithm b determines the unit of measure. The most common bases are $b = 2$ for *bits* and $b = e$ for *nats*. The information content is also called the *self-information* or *degree of surprise*. From Equ. 2.20, we observe that highly improbable realizations yield high information content and conversely for highly probable outcomes.

The expected value of the self-information is the *information entropy*, written as

$$H(X) \quad = \quad -\sum_{\Omega_X} p(X = x) \log_b p(X = x). \tag{2.21}$$

In relation to probability, the entropy is yet another measure of uncertainty about the realizations x of the random variable X. If all outcomes are equiprobable, the entropy is therefore maximum. From Equ. 2.21, Shannon derived the *noiseless coding theorem*, which provides a lower bound on the average number of bits required to transmit the state of a random variable. The rationale is that highly probable outcomes should be coded with shorter binary strings than lower probable ones in order to minimize the average amount of transmitted bits. The definition of Shannon's entropy is notably related to earlier work in statistical mechanics and has been subject to further publications by Jayne [Jay57].

Continuous Random Variables

Since the probability for a continuous random variable to take a specific value from its sample space is zero by definition, we need to extend the concept of entropy into the *differential entropy* as

$$h(X) \quad = \quad -\int_{\Omega_X} f_X(x) \log_b f_X(x) \, \mathrm{d}x. \tag{2.22}$$

The differential entropy is not a straightforward extension of the discrete entropy for the case of an infinite sample space and we thus denote it h. In particular, it can even take negative values and is not invariant under general transformations.

Information entropy can be generalized over a set of random variables or a multivariate random variable $\mathbf{X} = (X_1, \ldots, X_n)^T$ admitting a joint probability density function $f_{\mathbf{X}}$. In that case, it is called a *joint entropy* and is defined as

$$h(\mathbf{X}) = -\int_{\Omega_{\mathbf{X}}} f_{\mathbf{X}}(\mathbf{x}) \log_b f_{\mathbf{X}}(\mathbf{x}) \, \mathrm{d}\mathbf{x}. \tag{2.23}$$

Similarly, if we express the entropy of $\mathbf{X}$ given a particular outcome $\mathbf{z}$ of $\mathbf{Z}$ as

$$h(\mathbf{X} \mid \mathbf{Z} = \mathbf{z}) = -\int_{\Omega_{\mathbf{X}}} f_{\mathbf{X}}(\mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \log_b f_{\mathbf{X}}(\mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \, \mathrm{d}\mathbf{x}, \tag{2.24}$$

we can define the *conditional entropy* as the expectation of Equ. 2.24 with respect to $f_{\mathbf{Z}}$, *i.e.*,

$$h(\mathbf{X} \mid \mathbf{Z}) = \int_{\Omega_{\mathbf{Z}}} h(\mathbf{X} \mid \mathbf{Z} = \mathbf{z}) f_{\mathbf{Z}}(\mathbf{z}) \, \mathrm{d}\mathbf{z}. \tag{2.25}$$

Furthermore, we can formulate Equ. 2.25 as

$$h(\mathbf{X} \mid \mathbf{Z}) = \int_{\Omega_{\mathbf{Z}}} \int_{\Omega_{\mathbf{X}}} f_{\mathbf{X},\mathbf{Z}}(\mathbf{x}, \mathbf{z}) \log_b \frac{f_{\mathbf{Z}}(\mathbf{z})}{f_{\mathbf{X},\mathbf{Z}}(\mathbf{x}, \mathbf{z})} \, \mathrm{d}\mathbf{x} \, \mathrm{d}\mathbf{z}. \tag{2.26}$$

One of the most useful quantity in information theory is the *mutual information*. For two random variables $\mathbf{X}$ and $\mathbf{Z}$, it measures the average amount of information that can be obtained on $\mathbf{X}$ by observing an occurrence of $\mathbf{Z}$ and is defined as

$$I(\mathbf{X}; \mathbf{Z}) = \int_{\Omega_{\mathbf{Z}}} \int_{\Omega_{\mathbf{X}}} f_{\mathbf{X},\mathbf{Z}}(\mathbf{x}, \mathbf{z}) \log_b \frac{f_{\mathbf{X},\mathbf{Z}}(\mathbf{x}, \mathbf{z})}{f_{\mathbf{X}}(\mathbf{x}) f_{\mathbf{Z}}(\mathbf{z})} \, \mathrm{d}\mathbf{x} \, \mathrm{d}\mathbf{z}. \tag{2.27}$$

The mutual information quantifies the amount of uncertainty that remains on $\mathbf{X}$ if $\mathbf{Z}$ is known and is also a measure of the independence of the random variables. Indeed, $\mathbf{X}$ and $\mathbf{Z}$ are independent if and only if their mutual information is zero. Furthermore, the mutual information is symmetric, always non-negative, and well-defined for continuous and discrete random variables.

Like the mutual information, the *Kullback-Leibler divergence* [KL51] addresses the issues of the differential entropy and remains well-behaved for discrete and continuous random variables. For two random variables $\mathbf{X}$ and $\mathbf{Z}$ with identical sample space, it is defined by

$$\mathcal{D}_{KL}(\mathbf{X} \,||\, \mathbf{Z}) = \int_{\Omega_{\mathbf{X}}} f_{\mathbf{X}}(\mathbf{x}) \log \frac{f_{\mathbf{X}}(\mathbf{x})}{f_{\mathbf{Z}}(\mathbf{x})} \, \mathrm{d}\mathbf{x}. \tag{2.28}$$

The Kullback-Leibler divergence measures the *relative entropy* or *information gain* between $\mathbf{X}$ and $\mathbf{Z}$. If $\mathbf{Z}$ is used to approximate $\mathbf{X}$, it represents the information loss. It is also a non-symmetric distance between $\mathbf{X}$ and $\mathbf{Z}$.

2.1.3 Statistical Inference

Statistical inference aims at drawing conclusions about unobserved quantities, based on the observation of data subject to random variations. In a more general sense, statistics is considered to be an applied science that uses probability theory to manipulate and interpret data.

Statistical inference starts with the elaboration of a *probabilistic model* that describes the relations between the involved random variables. This model provides a joint probability density function of the random variables. Some of the random variables can be directly observed, in which case they are *measurable*, or inferred from other observed variables, in which case they are *latent* or *hidden*. To stick with statistical notation, the latent variable to be inferred is called a *parameter* and denoted $\boldsymbol{\Theta}$ or $\boldsymbol{\theta}$ for a specific value. Likewise, the measurable variable will be denoted $\mathbf{X}$ and one realization $\mathbf{x}$ is referred to as the *data*.

Bayes's Theorem

As stated previously, we initially define a joint probability density function $f_{\boldsymbol{\Theta},\mathbf{X}}$, which can be expressed in terms of conditional and marginal densities according to Equ. 2.12 as

$$f_{\boldsymbol{\Theta},\mathbf{X}}(\boldsymbol{\theta},\mathbf{x}) \quad = \quad f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}) f_{\boldsymbol{\Theta}}(\boldsymbol{\theta}). \tag{2.29}$$

Equ. 2.29 is commonly referred to as the *product rule* and its generalization to multiple variables in the joint density as the *chain rule of probability*. Furthermore, using the fact that $f_{\boldsymbol{\Theta},\mathbf{X}}(\boldsymbol{\theta},\mathbf{x}) = f_{\mathbf{X},\boldsymbol{\Theta}}(\mathbf{x},\boldsymbol{\theta})$, we can derive *Bayes's theorem* as

$$f_{\boldsymbol{\Theta}}(\boldsymbol{\theta} \mid \mathbf{X} = \mathbf{x}) \quad = \quad \frac{f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}) f_{\boldsymbol{\Theta}}(\boldsymbol{\theta})}{f_{\mathbf{X}}(\mathbf{x})}. \tag{2.30}$$

In Equ. 2.30, $f_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$ is known as the *prior* density, $f_{\boldsymbol{\Theta}}(\boldsymbol{\theta} \mid \mathbf{X} = \mathbf{x})$ as the *posterior* density, and $f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta})$ as the *data distribution*. For given observed data $\mathbf{x}$ and varying parameter $\boldsymbol{\theta}$, the data distribution can be interpreted as a *likelihood function* and denoted $\mathcal{L}(\boldsymbol{\theta} \mid \mathbf{x})$. The result of the evaluation of the likelihood function is called the *likelihood* of the parameter given the data. Since the data is actually observed, the denominator $f_{\mathbf{X}}(\mathbf{x})$ can be considered a constant and omitted to yield the unnormalized posterior density

$$f_{\boldsymbol{\Theta}}(\boldsymbol{\theta} \mid \mathbf{X} = \mathbf{x}) \quad \propto \quad f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}) f_{\boldsymbol{\Theta}}(\boldsymbol{\theta}), \tag{2.31}$$

which is commonly understood as the posterior density is proportional to the product of the prior density and the likelihood, or simply

$$\text{posterior} \quad \propto \quad \text{likelihood} \times \text{prior}. \tag{2.32}$$

Bayes's theorem unveils the core of statistical inference. Starting from the joint density of an unknown parameter and measurable data, we can derive the posterior density of that parameter when the data is observed.

Data Sample

In a real-world scenario, the data generally consists in multiple *measurements* of some physical quantity of interest. In probabilistic terms, we perform n random experiments in which a measurable random variable $\mathbf{X}$ is observed. This results in a sequence of random variables $\mathbf{X}_1, \ldots, \mathbf{X}_n$, a *data sample*, whose realizations $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are the observed data. A sample is usually assumed to be *exchangeable*, *i.e.*, any permutation of its indices yields the same joint density. Furthermore, if each variable $\mathbf{X}_i$ in the sample follows the same probability distribution as $\mathbf{X}$, they are *identically distributed* (abbreviated *i.d.*). Lastly, if the $\mathbf{X}_i$ are identically distributed *and* mutually independent, they are *independent and identically distributed* (abbreviated *i.i.d.*). For a sample of size n, the *sample mean* is a random variable defined as

$$\mathbf{S}_n = \frac{1}{n}\sum_{i=1}^{n}\mathbf{X}_i, \tag{2.33}$$

whose realization is denoted $\mathbf{s}_n$. Furthermore, if the sample is *i.i.d.*, it is straightforward to derive the first raw moment and second central moment of $\mathbf{S}_n$ as

$$\begin{aligned}\mathbb{E}\left[\mathbf{S}_n\right] &= \mathbb{E}\left[\mathbf{X}\right] \\ \mathrm{Var}\left[\mathbf{S}_n\right] &= \frac{1}{n}\mathrm{Var}\left[\mathbf{X}\right].\end{aligned} \tag{2.34}$$

The *sample covariance* matrix is defined as

$$\mathbf{S}_n^2 = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{X}_i - \mathbf{S}_n)(\mathbf{X}_i - \mathbf{S}_n)^T. \tag{2.35}$$

and its first raw moment is

$$\mathbb{E}\left[\mathbf{S}_n^2\right] = \frac{n-1}{n}\mathrm{Var}\left[\mathbf{X}\right]. \tag{2.36}$$

The concept of sample is not limited to the measurement of physical quantities. Indeed, we can *simulate* a sample using the technique of *pseudo-random number* generation. Most modern computers are endowed with a pseudo-random generator that can produce a sequence of random variates for the majority of probability distributions. We are thus able to draw samples of any kind of random variables, latent or measurable. The advent of this technique has largely contributed to the popularization of Bayesian methods in statistics [GCSR03]. Indeed, Bayesian methods often involve integrals, for instance in the marginalization, that can rarely be solved analytically.

Parametric Probability Density Functions

Probability density functions are commonly represented by a *parametric model*, *i.e.*, a family of functions that can be described with a finite number of parameters. A parametric *family of functions* is a set of functions, involving one or

more parameters and the argument, that share the same formula. For a random variable $\mathbf{X}$ admitting a probability density function $f_{\mathbf{X}}$, its parameter will be considered a random variable that will be denoted $\boldsymbol{\Theta}_{\mathbf{X}}$ and its random variate $\boldsymbol{\theta}_{\mathbf{X}}$. The family of functions describing the density will be written $f(\mathbf{x}; \boldsymbol{\theta}_{\mathbf{X}})$. If we want to emphasize the probabilistic interpretation, a *family of distributions* will be denoted as $f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\Theta}_{\mathbf{X}})$. The most dominant class of parametric models is the *exponential family*. Interestingly, it contains many probability distributions that reflect real physical phenomena and have useful algebraic properties. We shall briefly introduce the main families of distributions used in this thesis in the remainder of this section.

The *normal or Gaussian distribution* is represented by a probability density function governed by two parameters, namely the *mean* $\boldsymbol{\mu}$ and the *covariance matrix* $\boldsymbol{\Sigma}$. These two parameters correspond respectively to the first raw moment and the second central moment of the density. Although the normal distribution can be formulated for univariate random variables, we shall, as previously, focus on the more general multivariate setting. We formulate the probability density function of a K-dimensional normally distributed random vector $\mathbf{X}$ as

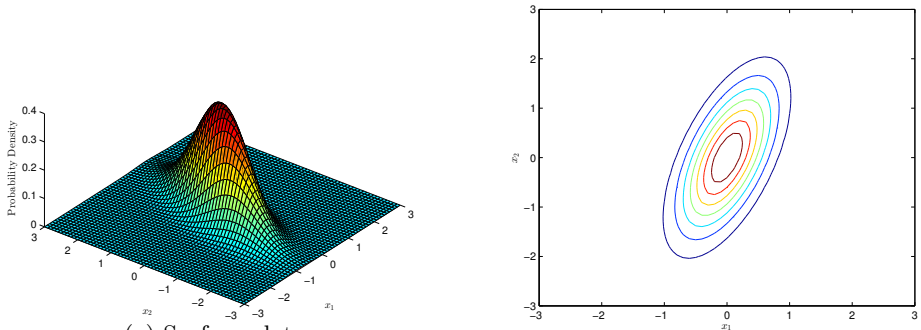
$$f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{K/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}. \tag{2.37}$$

The inner part of the exponential can be expressed as a squared *Mahalanobis distance*

$$D^2 = (\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}). \tag{2.38}$$

The normal density function, an example thereof is depicted in Fig. 2.1, will be denoted as $\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$. In addition, we shall use the notation $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to state that $\mathbf{X}$ is normally distributed. A *standard normal distribution* has zero mean and identity covariance matrix. The prevalence of the normal distribution in physical phenomena has been historically justified by the *central limit theorem*. In its classical form, it states that for a large n the sample mean $\mathbf{S}_n$ of an *i.i.d.* sample of $\mathbf{X}$ approximatively follows a normal distribution, *i.e.*, $\mathbf{S}_n \approx \mathcal{N}(\mathbb{E}[\mathbf{X}], \frac{1}{n}\mathrm{Var}[\mathbf{X}])$. The *i.d.* assumption can even be relaxed in a more general formulation under some conditions. Intuitively, the measurement of a physical phenomenon can often be considered as the sum of many different random effects and therefore tends to follow a normal distribution. Maxwell and Boltzmann have also demonstrated that particles' velocities at thermodynamic equilibrium follow a normal distribution. The close relation between thermodynamic entropy and information entropy has led to the *principle of maximum entropy* [Jay57], which guides the selection of a probability distribution under a set of constraints. Under this principle, the normal distribution maximizes the entropy for a given mean and variance, *i.e.*, it is the most conservative distribution or the one which makes the fewest assumptions.

The normal distribution has several appealing properties [Bis06]. Let us define a normally distributed random variable $\mathbf{X} = (X_1, \ldots, X_K)^T$ and a partition such that $\mathbf{X}_a = (X_1, \ldots, X_D)^T$ and $\mathbf{X}_b = (X_{D+1}, \ldots, X_K)^T$. In terms of the parameters, the partition yields the mean



(a) Surface plot.

(b) Contour plot.

Figure 2.1: Probability density function of a bivariate normal distribution represented as a surface plot (Fig. 2.1a) and a contour plot (Fig. 2.1b). The elliptic shape of the contour lines is governed by the covariance matrix; the variance elements on the main diagonal affect the ellipse's radii ($\mathrm{Var}[X_1] < \mathrm{Var}[X_2]$ in this example) and the off-diagonal elements define a rotation of the ellipse around its center.

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix} \tag{2.39}$$

and the covariance matrix

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{pmatrix}. \tag{2.40}$$

The marginal density of $\mathbf{X}_a$ is a normal distribution with mean $\boldsymbol{\mu}_a$ and covariance $\boldsymbol{\Sigma}_{aa}$, *i.e.*, $\mathbf{X}_a \sim \mathcal{N}(\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_{aa})$. The conditional density of $\mathbf{X}_a$ given the occurrence $\mathbf{x}_b$ of $\mathbf{X}_b$ is also a normal distribution with mean

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\mathbf{x}_b - \boldsymbol{\mu}_b) \tag{2.41}$$

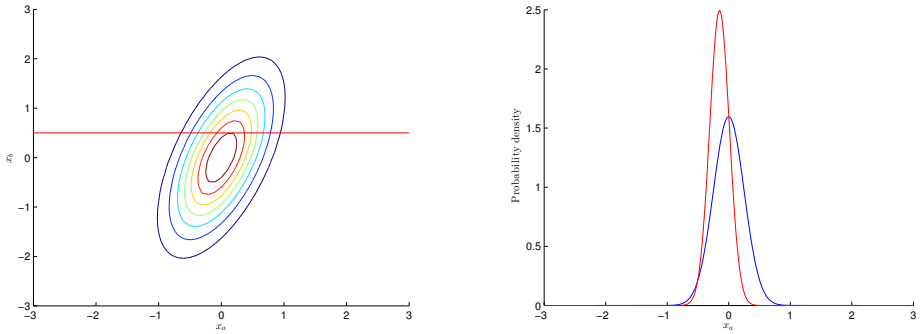
and covariance matrix given by the *Schur complement* of Equ. 2.40

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}. \tag{2.42}$$

We illustrate the marginal and conditional distributions of a bivariate normal distribution in Fig. 2.2.

The *chi-squared distribution* is characterized by a probability density function defined over univariate random variates and is governed by a single parameter k, known as *degrees of freedom*. The mathematical definition of the density for a random variable X is

$$f_X(x \mid k) = \begin{cases} \frac{x^{\frac{k}{2}-1}\exp\left(-\frac{x}{2}\right)}{2^{\frac{k}{2}}\Gamma\left(\frac{k}{2}\right)} & \text{for } x \geq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{2.43}$$



(a) Joint pdf. (b) Marginal and conditional pdf.

Figure 2.2: Properties of a bivariate normal distribution. Fig. 2.2b shows the marginal distribution $f_{\mathbf{X}_a}(x_a)$ (blue curve) and the conditional distribution $f_{\mathbf{X}_a}(x_a \mid x_b = 0.5)$ (red curve) of the bivariate normal distribution in Fig. 2.2a.

where $\Gamma(\cdot)$ is the *gamma function*. The chi-squared density function, displayed in Fig. 2.3 for $k = 3$, will be denoted $\chi^2(x \mid k)$ and a random variable X following a chi-squared distribution $X \sim \chi^2(k)$. By definition, a random variable which consists in the sum of k squared independent standard normal random variables is chi-squared distributed. Furthermore, the squared Mahalanobis distance of a data sample of a K-dimensional normally distributed random variable follows a chi-squared distribution with K degrees of freedom. Therefore, the chi-squared distribution provides a simple method for detecting *outliers* in multivariate data. Indeed, data points with squared Mahalanobis distance larger than a given quantile (*e.g.* $p = 0.975$) can be declared as outliers. This property will be subject to further discussions in Sec. 2.1.5. More generally, this distribution is widely used for *statistical hypothesis testing* in frequentist inference [Pea00]. The chi-squared distribution can finally be interpreted as a special case of the *gamma distribution* which serves as a prior density in many Bayesian inference problems.

The *uniform distribution* is defined for discrete or continuous random variables and has an extension to the multivariate setting. It is characterized by a *support* region, *i.e.*, a subset of the sample space where all outcomes are equiprobable. In the univariate continuous case, the support consists in an interval $[a, b]$ and the density is

$$f_X(x \mid a, b) = \begin{cases} \frac{1}{b-a} & \text{for } x \in [a, b] \\ 0 & \text{otherwise.} \end{cases} \tag{2.44}$$

The uniform density with support $[a, b]$, an example thereof is shown in Fig. 2.4, will be denoted $\mathrm{U}(x \mid a, b)$ and a random variable associated with a uniform distribution $X \sim \mathrm{U}(a, b)$. The *standard uniform distribution* is a uniform distribution with $a = 0$ and $b = 1$. As long as the probability density function integrates to 1, Equ. 2.44 generalizes to multiple dimensions or to discrete random variables. Unlike the previously presented distributions, the uniform density is not a member of the exponential family. According to the principle of maximum entropy, the uniform distribution is the distribution with maximum entropy when no constraints are assumed. Historically, it was coined the *principle of*

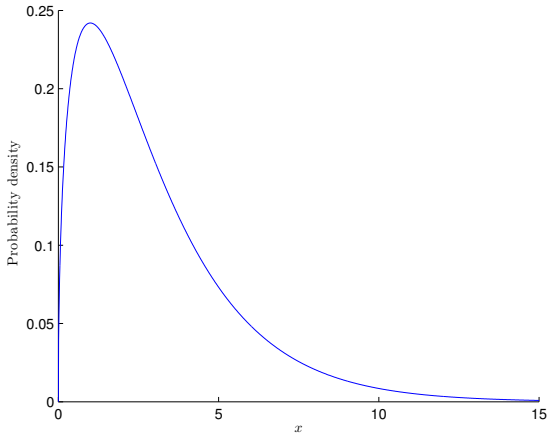


Figure 2.3: Probability density function of a chi-squared distribution with $k = 3$ degrees of freedom. The chi-squared distribution is the distribution of a sum of k squared independent standard normal random variables or the distribution of the squared Mahalanobis distance of a data sample of a k-dimensional normally distributed random vector.

insufficient reason or the *principle of indifference* in Laplace's work. In thermodynamics, it also corresponds to the distribution of the microstates at the equilibrium, *i.e.*, when the system reaches its maximum entropy (from the second law of thermodynamics). The standard uniform distribution also plays a crucial role in random variates generators for arbitrary distributions using the technique of *inverse transform sampling* [Dev86]. Apart from its role in the assignment of a distribution with maximum entropy, the uniform distribution also appears in the spatial arrangement of natural species.

Choice of Prior

After the presentation of some common probability distributions, we shall conclude this section with a discussion on the choice of prior distributions appearing in any statistical inference problem. As the name suggests, a prior should encapsulate the *a priori* knowledge of an expert before considering any data and in essence be entirely subjective. This type of prior is generally coined an *informative prior*. In many applications, however, only vague knowledge is accessible and an *uninformative prior* is preferred, *e.g.*, a uniform distribution. In contrast to informative priors, uninformative priors are considered objective. The principle of maximum entropy, therefore of minimum information, provides a justification and a methodology for choosing this type of prior. For convenience, an uninformative prior is frequently expressed in the form of an *improper prior* which violates the conditions in Equ. 2.1. However, when suitably chosen, the posterior density will become proper and therefore justifies its use. For instance, although a uniform density is not defined on the entire real line, a similar concept can be conveyed through an improper prior. The parameters of a prior probability density function are called *hyperparameters*. In many situations, the prior is subjectively chosen for mathematical convenience to be a *conjugate prior* to the likelihood function. A conjugate prior provides a closed-form ex-

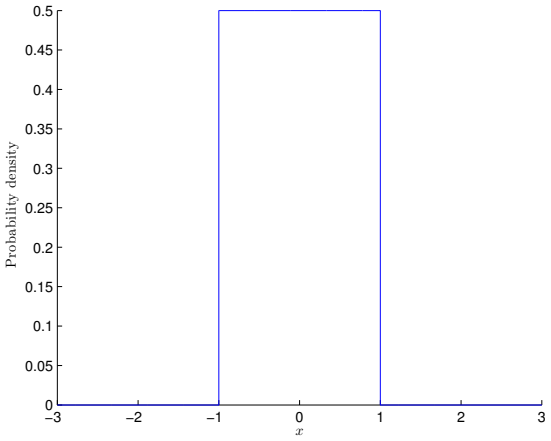


Figure 2.4: Probability density function of a univariate continuous uniform distribution with support $[a = -1, b = 1]$. All outcomes within the support region are equiprobable.

pression for the posterior density function in the same family of distributions. All members of the exponential family admit conjugate priors and their posterior computation consists in updating their hyperparameters. In the case of an *i.i.d.* sample, Bayes' theorem is amenable to sequential inference, *i.e.*, the posterior can be computed with one data point at the time. This posterior becomes thus the prior for the next data point. Intuitively, the more data points are used for updating the posterior distribution, the less the initial prior will have an influence.

2.1.4 Statistical Estimation

Given a probabilistic model and a data sample, the Bayesian framework established thus far provides the full knowledge about an unknown parameter in the form of a posterior probability density function. Most of the real-world scenarios, however, require the selection of a single value to be assigned to the parameter. As a prolongation of probabilistic inference, this problem can be considered as a topic of *decision theory*, *i.e.*, the selection of the optimal value for the parameter with respect to a given *loss function* or *cost function*. Frequentist statisticians have historically tackled this problem under the precepts of *estimation theory*. Estimation theory has naturally arisen from experimental sciences where researchers were repetitively measuring physical quantities and trying to output the best value out of these measurements. The rest of this section shall review the main methods of estimation and demonstrate the close link to Bayesian decision theory.

Frequentist Estimation

Classical estimation theory is mainly concerned with the elaboration of an *estimator* $\widehat{\boldsymbol{\Theta}}$ for an unknown parameter $\boldsymbol{\Theta}$ of interest and the characterization of its properties. From a mathematical point of view, an estimator is a function of a data sample and is therefore itself a random variable with an associated

sampling distribution. From a given observed data sample, the estimator yields an *estimate* $\widehat{\boldsymbol{\theta}}$ for the parameter. This latter is commonly referred to as the *estimand*. The frequentist paradigm assumes that the unknown parameter is deterministic [Ney37] and we denote its true value as $\boldsymbol{\theta}$. As any function of the data can be an estimator, we need to establish a couple of properties to characterize its quality. For a data sample $\mathbf{X}$, the error of the estimator is defined as

$$e(\widehat{\boldsymbol{\Theta}}) \quad = \quad \widehat{\boldsymbol{\Theta}} - \boldsymbol{\theta}. \tag{2.45}$$

The *mean-squared error (MSE)* of the estimator is

$$\mathrm{MSE}(\widehat{\boldsymbol{\Theta}}) \quad = \quad \mathbb{E}_{\mathbf{X}}\left[e(\widehat{\boldsymbol{\Theta}})^T e(\widehat{\boldsymbol{\Theta}})\right]. \tag{2.46}$$

The mean-squared error indicates the average squared error over all possible estimates. The *bias* of an estimator is defined as

$$B(\widehat{\boldsymbol{\Theta}}) \quad = \quad \mathbb{E}_{\mathbf{X}}\left[\widehat{\boldsymbol{\Theta}}\right] - \boldsymbol{\theta}. \tag{2.47}$$

If $B(\widehat{\boldsymbol{\Theta}}) = 0$, the estimator is *unbiased* and this is usually a desirable property, along with a low variance and mean-squared error. The variance, bias, and mean-squared error are further related through the following equation

$$\mathrm{MSE}(\widehat{\boldsymbol{\Theta}}) \quad = \quad \mathrm{tr}\left(\mathrm{Var}\left[\widehat{\boldsymbol{\Theta}}\right]\right) + B(\widehat{\boldsymbol{\Theta}})^T B(\widehat{\boldsymbol{\Theta}}), \tag{2.48}$$

where $\mathrm{tr}\,(\cdot)$ denotes the *trace* operator. An estimator is *consistent* if it converges in probability to the true parameter for an infinite sample size, *i.e.*, the estimates $\widehat{\boldsymbol{\theta}}$ become arbitrarily close to $\boldsymbol{\theta}$.

If we denote the data distribution as $f(\mathbf{x}; \boldsymbol{\theta})$, the *score* is defined as the partial derivative with respect to $\boldsymbol{\theta}$ of the logarithm of the density function, *i.e.*,

$$\mathcal{V}(\boldsymbol{\theta}, \mathbf{X}) \quad = \quad \frac{\partial}{\partial \boldsymbol{\theta}} \log f(\mathbf{x}; \boldsymbol{\theta}). \tag{2.49}$$

The expected value of the score over $\mathbf{X}$ is analytically zero and its variance defines the *Fisher information*

$$\mathcal{I}(\boldsymbol{\theta}) \quad = \quad \mathrm{Var}\left[\mathcal{V}(\boldsymbol{\theta}, \mathbf{X})\right]. \tag{2.50}$$

In the multivariate setting, Equ. 2.50 is referred to as the *Fisher information matrix (FIM)*. Intuitively, the Fisher information measures the amount of information carried by the measurable random variable $\mathbf{X}$ about the unknown parameter $\boldsymbol{\theta}$. Although related through the Kullback-Leibler divergence, Fisher information ought not to be confused with Shannon information. While the former quantifies how well we can estimate an unknown parameter with the

data, the latter measures the number of bits required to transmit the state of a random variable. In Bayesian statistics, Fisher information is also used to build uninformative priors, namely *Jeffreys priors* [Jef46], and appears in the Laplace approximation of posterior densities [GCSR03].

When devising an estimator, some criteria of quality and optimality are required. Amongst them, we might be interested in unbiased estimators with minimal variance. These estimators are known as *minimum-variance unbiased (MVU) estimators*. In that context, the *Cramér-Rao bound* [Rao45] provides the minimal achievable variance as

$$\operatorname{Var}\left[\widehat{\boldsymbol{\Theta}}\right] \quad \geq \quad \mathcal{I}^{-1}(\boldsymbol{\theta}), \tag{2.51}$$

where, in the multivariate case, the matrix inequality simply states that the difference between the variance and the inverse of the Fisher information matrix is *positive semidefinite*. If the variance of the estimator attains the bound, it is coined *efficient*. By Equ. 2.48, an MVU estimator also minimizes the mean-squared error. If we are not restricted to the class of unbiased estimators, another sought optimization criterion is minimal mean-squared error, yielding the *minimum mean-squared error (MMSE) estimator*. In many practical applications, optimality criteria only apply when the sample size grows towards infinity. In that case, we shall be interested in the *asymptotic behavior* of the estimator, *e.g.*, asymptotic efficiency. Finite-sample efficiency is indeed only possible for distribution parameters of the exponential family.

Maximum-Likelihood Estimator

Under the frequentist paradigm, the most popular method for building an estimator is to maximize the likelihood function from Equ. 2.30, resulting in the *maximum-likelihood (ML) estimator*. The rationale behind the maximum-likelihood method is that the sought parameter will optimally fit the chosen model to the data. Although Fisher [Fis22] advocated the use of the maximum-likelihood principle, it dates back to the work of Gauss on the least-squares problem. Formally, the most common formulation starts with an *i.i.d.* data sample $\mathbf{X}_1, \cdots, \mathbf{X}_n$ from a measurable random variable $\mathbf{X}$ and a data distribution $f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta})$ from which we want to estimate $\boldsymbol{\theta}$. Due to the *i.i.d* assumption, the $\mathbf{X}_i$ are conditionally independent given the occurrence $\boldsymbol{\theta}$ of $\boldsymbol{\Theta}$ and their joint data distribution can be factorized into

$$f_{\mathbf{X}_1,\cdots,\mathbf{X}_n}(\mathbf{x}_1, \cdots, \mathbf{x}_n \mid \boldsymbol{\Theta} = \boldsymbol{\theta}) \quad = \quad \prod_{i=1}^{n} f_{\mathbf{X}}(\mathbf{x}_i \mid \boldsymbol{\Theta} = \boldsymbol{\theta}). \tag{2.52}$$

The left-hand side of Equ. 2.52 is then considered as a likelihood function denoted $\mathcal{L}(\boldsymbol{\theta} \mid \mathbf{x}_1, \cdots, \mathbf{x}_n)$ with fixed data and variable parameter. For mathematical and numerical convenience, it is common to employ the logarithm of the likelihood function and formulate the ML estimator as

$$\widehat{\boldsymbol{\Theta}}_{ML} \quad = \quad \operatorname*{argmax}_{\boldsymbol{\theta}} \log \mathcal{L}(\boldsymbol{\theta} \mid \mathbf{x}_1, \cdots, \mathbf{x}_n). \tag{2.53}$$

From Equ. 2.53, it is clear that the search for an estimator has been formulated as a *mathematical optimization problem* where the log-likelihood function is interpreted as an *objective function*. By convention, the *maximization problem* is turned into the *minimization problem*

$$\widehat{\boldsymbol{\Theta}}_{ML} \quad = \quad \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \log \mathcal{L}(\boldsymbol{\theta} \mid \mathbf{x}_1, \cdots, \mathbf{x}_n), \tag{2.54}$$

where the negative log-likelihood is now a *cost function* or *loss function*. Furthermore, if the factorization in Equ. 2.52 applies, we end up with the classical expression

$$\widehat{\boldsymbol{\Theta}}_{ML} \quad = \quad \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{n} - \log f_{\mathbf{X}}(\mathbf{x}_i \mid \boldsymbol{\Theta} = \boldsymbol{\theta}). \tag{2.55}$$

Although there might exist a closed-form solution to Equ. 2.54 depending on the form of the likelihood function, numerical techniques are often required. These methods can generally not distinguish a *local minimum* from a *global minimum* and there might not even exist a solution. For finite sample sizes, the ML estimator does not satisfy any optimality criteria. However, when the sample size grows to infinity, we can state several asymptotic properties. These properties are studied within the *large sample theory* or *asymptotic theory* and are mostly justified by the *law of large numbers* and the *central limit theorem*. Under certain conditions on the likelihood function, the ML estimator is consistent, *asymptotically efficient*, *asymptotically normally* distributed, and therefore an MVU estimator. As a consequence, the ML estimator converges in distribution to

$$\widehat{\boldsymbol{\Theta}}_{ML} \quad \xrightarrow{d} \quad \mathcal{N}\left(\boldsymbol{\theta}, \frac{\mathcal{I}^{-1}(\boldsymbol{\theta})}{n}\right). \tag{2.56}$$

The maximum-likelihood method is not limited to the common factorization in Equ. 2.52, but generalizes to any problem where a joint density of the data can be formulated. However, in that case, the appealing asymptotic properties of the ML estimator do not hold.

Maximum a Posteriori Estimator

The maximum-likelihood framework is attractive in the sense that it allows for the estimation of model parameters without dealing with the complexity of Bayesian theory. However, it is not suitable to prevent overfitting or solve ill-posed problems. For that reasons, it can be extended by adding a *regularization term*. Under the Bayesian viewpoint, this technique boils down to estimating the mode of the posterior distribution of the parameter, which yields the *maximum a posteriori (MAP) estimator*. Formally, the MAP estimator is found by maximizing the left-hand side of Equ. 2.30, *i.e.*,

$$\widehat{\boldsymbol{\Theta}}_{MAP} \quad = \quad \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, f_{\boldsymbol{\Theta}}(\boldsymbol{\theta} \mid \mathbf{X}), \tag{2.57}$$

where we have not assigned a variate to $\mathbf{X}$ to keep the estimator a function of random variables. If the posterior is available in closed-form (*e.g.* using a conjugate prior), there is an analytical solution to Equ. 2.57. Otherwise, we resort to numerical optimization or sampling techniques. In general, MAP estimation does not require to work out the posterior distribution. Indeed, as the denominator in Equ. 2.30 does not depend on the parameter, we can express the MAP estimator as

$$\widehat{\boldsymbol{\Theta}}_{MAP} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}) f_{\boldsymbol{\Theta}}(\boldsymbol{\theta}). \tag{2.58}$$

Furthermore, when using the same assumptions as in Equ. 2.52, we can interpret Equ. 2.58 as for the ML estimator and transform it into

$$\widehat{\boldsymbol{\Theta}}_{MAP} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[\sum_{i=1}^{n} -\log f_{\mathbf{X}}(\mathbf{x}_i \mid \boldsymbol{\Theta} = \boldsymbol{\theta}) - \log f_{\boldsymbol{\Theta}}(\boldsymbol{\theta}) \right]. \tag{2.59}$$

If we use an improper uniform prior in Equ. 2.58, the MAP estimator becomes an ML estimator. Furthermore, from the large sample theory point of view, the influence of the prior asymptotically vanishes and the MAP estimator therefore similarly converges to the normal distribution

$$\widehat{\boldsymbol{\Theta}}_{MAP} \xrightarrow{d} \mathcal{N}\left(\boldsymbol{\theta}, \frac{\mathcal{I}^{-1}(\boldsymbol{\theta})}{n}\right). \tag{2.60}$$

Bayes Estimator

Thus far, we have derived the most important properties of estimation theory and presented two popular methods for constructing an estimator. Nevertheless, it can prove truly enlightening to consider the *Bayesian decision theory* perspective [Ber85]. As probabilistic inference relates to reasoning under uncertainty, decision theory tackles the problem of choice under uncertainty. Indeed, an estimator can be interpreted as a *decision rule* under a certain *loss function*. In that context, the loss function is a function of the estimand $\boldsymbol{\Theta}$ and an estimator $\widehat{\boldsymbol{\Theta}}$, which we denote $L(\boldsymbol{\Theta}, \widehat{\boldsymbol{\Theta}})$, and a *Bayes estimator* attempts to minimize the posterior expected loss for each realization $\mathbf{x}$ of $\mathbf{X}$, *i.e.*,

$$\widehat{\boldsymbol{\Theta}}_{Bayes} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \mathbb{E}_{\boldsymbol{\Theta}|\mathbf{X}}\left[L(\boldsymbol{\Theta}, \widehat{\boldsymbol{\Theta}})\right], \tag{2.61}$$

where the expectation is taken over the posterior density $f_{\boldsymbol{\Theta}}(\boldsymbol{\theta} \mid \mathbf{X})$. The most common loss function is the *quadratic loss function* with

$$L(\boldsymbol{\Theta}, \widehat{\boldsymbol{\Theta}}) = (\widehat{\boldsymbol{\Theta}} - \boldsymbol{\Theta})^T (\widehat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}). \tag{2.62}$$

We can note the close similitude between Equ. 2.62 and Equ. 2.46, the main difference being that under the Bayesian setting the unknown parameter is a random variable. In that case, the mean-squared error can be obtained by

taking the expectation of Equ. 2.62 over the joint density of $\mathbf{X}$ and $\boldsymbol{\Theta}$ and is referred to as the *Bayes risk*. A Bayes estimator minimizes the Bayes risk and thus, under the quadratic loss function, the resulting estimator is an MMSE estimator. Furthermore, the Bayes estimate coincides with the posterior mean of the parameter, *i.e.*,

$$\widehat{\boldsymbol{\Theta}}_{MMSE} = \mathbb{E}_{\boldsymbol{\Theta}|\mathbf{X}}[\boldsymbol{\Theta}]. \tag{2.63}$$

Alternatively, we can select the *0-1 loss function*

$$L(\boldsymbol{\Theta}, \widehat{\boldsymbol{\Theta}}) = \begin{cases} 0 & \text{if } \widehat{\boldsymbol{\Theta}} = \boldsymbol{\Theta} \\ 1 & \text{otherwise.} \end{cases} \tag{2.64}$$

Under this loss function, the Bayes estimator actually becomes an MAP estimator. The Bayes estimator is therefore a generalization of the ML and MAP estimators and benefits from the same asymptotic properties under the above loss functions. Through these derivations, we have thus revealed the tight interplay between Bayesian and frequentist estimation. For large samples or for uninformative priors, the posterior density of the parameter often corresponds to the sampling distribution of a frequentist estimator.

2.1.5 Robust Estimation

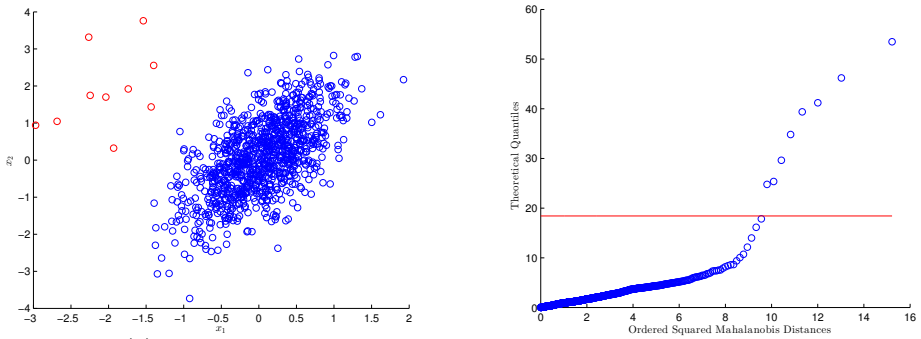
In the estimation framework presented thus far, we have silently assumed that the designed statistical model is a perfect representation of the data generation process. However, in real-world problems, some physical measurements, referred to as *outliers*, may deviate from this assumption. Outliers can have a real physical origin (*e.g.*, spurious measurement errors from a faulty sensor) or simply reflect a flaw in the statistical model (*e.g.*, data comes from a *mixture model* instead of a single distribution). Robust estimation is mainly concerned with the elaboration of efficient methods that can cope with outliers in the data to arrive at a *robust estimator*. We shall review some of the most popular approaches in the rest of this section.

Manual Screening and Mahalanobis Distance Thresholding

The most ancient method to deal with outliers is to manually inspect a *scatter plot* of the data and remove them from the estimation process. However, for large datasets or multivariate data, this task becomes increasingly difficult and tedious. As formerly presented in Sec. 2.1.3, for normally distributed data, the chi-squared distribution offers an automated way for defining a boundary between *inliers* and outliers. As shown in Fig. 2.5, by defining a certain level of confidence, outliers can thus be systematically discarded from the data.

Adaptation of the Data Distribution

Excluding data points from the estimation process remains a controversial approach and a modern strategy would rather focus on the improvement of the underlying statistical model to accommodate for the outliers. For the reasons outlined in Sec. 2.1.3, the normal distribution is a common choice for modeling



(a) Scatter plot. (b) Chi-squared plot.

Figure 2.5: Example of outlier detection and rejection in bivariate data. One thousand samples of an inlier bivariate normal distribution (blue circles) and ten samples of an outlier distribution (red circles) are depicted in the scatter plot in Fig. 2.5a. The ordered squared Mahalanobis distances of the samples are plotted against the theoretical quantiles of the chi-squared distribution in Fig. 2.5b. In this example, the red line represents the chi-squared quantile at $p = 0.9999$ and draws a clear boundary between inliers and outliers.

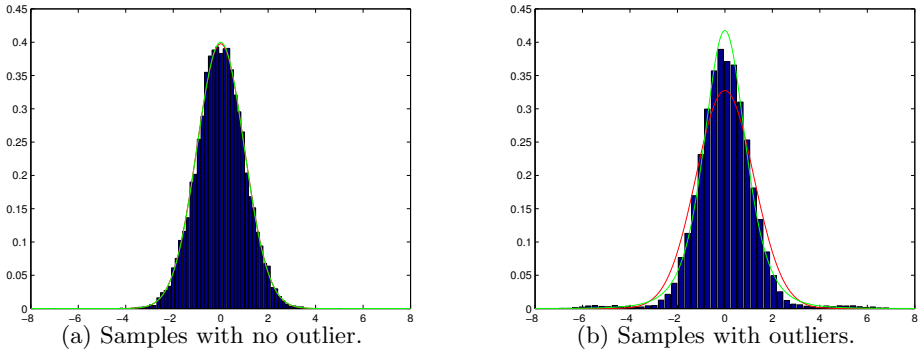
a data distribution and is also preferred for mathematical convenience. However, the normality assumption might not hold in practice and a heavy-tailed distribution such as the *Student's t-distribution* or the *Cauchy distribution* is more appropriate. As depicted in Fig. 2.6, these data distributions assign a higher likelihood to the data points appearing farther away from their central tendency and therefore naturally account for outliers. An example of this approach is presented in [BS04] in the context of robust mixture modeling.

Clustering and Mixture Modeling

If the inlier and outlier data points form distinct clusters of data, clustering techniques such as the *k-means* algorithm can provide an automated way for detecting outliers, as displayed in Fig. 2.7. In a probabilistic setting, the data distribution can also be modeled as a finite mixture model [AW80], using the *expectation-maximization (EM)* algorithm for estimation, or an infinite mixture model [SS11], with a *Dirichlet process mixture (DPM)* prior and sampling techniques for estimation. The problem of background modeling in computer vision [SG00] offers an example where this method is applicable. In this context, while the inlier data points consist in measurements of the background, dynamic objects can be interpreted as outliers.

M-Estimators

We have shown in Sec. 2.1.4 that an ML estimator can be formulated as the minimization of a sum of cost functions. If the data is assumed to be normally distributed, the problem becomes the minimization of a sum of *quadratic cost functions*, namely the squared Mahalanobis distances, *i.e.*,



(a) Samples with no outlier. (b) Samples with outliers.

Figure 2.6: Density estimation with robust data distribution. The red curve represents the maximum-likelihood estimate for the data sample using a normal distribution and the green curve using a Student's t-distribution. When the sample contains no outlier (Fig. 2.6a), the normal and Student's t-distributions provide similar estimates. The Student's t-distribution estimate is less affected by outliers (Fig. 2.6b).

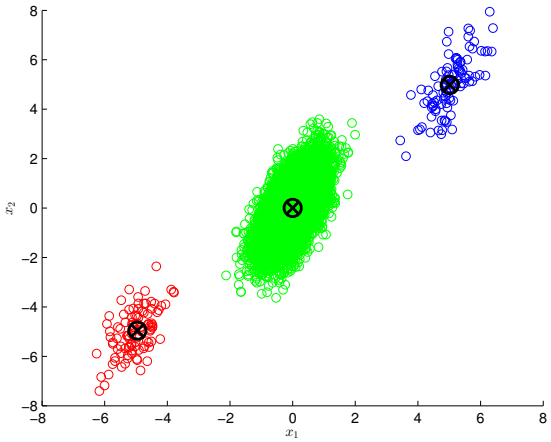


Figure 2.7: Example of the k-means clustering technique for detecting outliers. The inlier cluster (green circles) is clearly distinguishable from the two outlier clusters (red and blue circles.)

$$\widehat{\boldsymbol{\Theta}}_{ML} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{n} D_i^2. \tag{2.65}$$

From Equ. 2.65, we can recognize the formulation of a *least-squares problem*, which shall be the subject of further investigation in the next section. A quadratic cost function is naturally not robust to outliers. Indeed, a single outlier can cause an arbitrarily large error in the estimate. Nevertheless, this ML estimator can be transformed into a robust estimator by choosing a different form of cost function. For instance, if the sought parameter is the mean of a normal distribution, Equ. 2.65 yields the *sample mean* and when using an *absolute error cost function*, the estimator becomes the *sample median*. The sample median can be considered as more robust estimator for the location of the distribution in the sense that outliers will not dramatically change the estimates.

In that context, Huber [Hub64] developed the concept of *M-estimators* ("M" for "maximum likelihood-type") which are a generalization of maximum-likelihood estimators. The theory of M-estimators provides an appealing method to build robust estimators by converting the problem in Equ. 2.65 into an *iteratively reweighted least-squares (IRLS) problem*. The method starts by defining a symmetric and positive-definite cost function $\rho(D_i)$ and stating the problem as

$$\widehat{\boldsymbol{\Theta}}_{M} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho(D_i). \tag{2.66}$$

If ρ is differentiable and $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)^T$, the estimator can be expressed as the solution to the K following gradient equations

$$\textstyle\sum_{i=1}^{n} \psi(D_i) \frac{\partial D_i}{\partial \theta_j} = 0, \quad \text{for } j = 1, \ldots, K, \tag{2.67}$$

where

$$\psi(D_i) = \frac{\partial \rho(D_i)}{\partial \theta_j} \tag{2.68}$$

is referred to as the *influence function*. If we define the *weight function* as

$$w(D_i) = \frac{\psi(D_i)}{D_i}, \tag{2.69}$$

we can express Equ. 2.67 as

$$\textstyle\sum_{i=1}^{n} w(D_i) D_i \frac{\partial D_i}{\partial \theta_j} = 0, \quad \text{for } j = 1, \ldots, K, \tag{2.70}$$

which corresponds to the gradient equations of the following IRLS problem

$$\widehat{\boldsymbol{\Theta}}_{IRLS}^{(t)} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{n} w(D_i^{(t-1)}) D_i^2. \tag{2.71}$$

The IRLS algorithm uses the current estimate to compute a weight and, at each step, amounts to solving a *weighted least-squares problem*, which turns out to be a key property of the M-estimators. Indeed, as we shall present in the next section, least-squares problems have well-understood analytical solutions. Furthermore, the IRLS method is compatible with other iterative optimization methods such as *Gauss-Newton* or *Levenberg-Marquardt* algorithms.

Depending on the nature of the data, various forms of cost functions have been proposed. Apart from the quadratic and absolute error forms, we can mention the *Cauchy*, *Huber* [Hub81], or *Geman-McClure* [GM87] cost functions. For our application, we want to tailor the cost function in such a way that it behaves like the quadratic cost function for inliers and downweights the outliers. For this purpose, we advocate the *modified Blake-Zisserman* [HZ03] cost function, which takes the form

$$\rho(D) = -\log\left[\exp(-D^2) + \epsilon\right], \tag{2.72}$$

where ϵ is a parameter. The resulting weight function is

$$w(D) = \frac{\exp(-D^2)}{\exp(-D^2) + \epsilon}. \tag{2.73}$$

To satisfy our conditions, we have to choose the ϵ that yields a weight close to 1 for inliers and close to 0 for outliers. To this end, we can, as mentioned earlier, use the fact that the squared Mahalanobis distances are chi-squared distributed and search the quantile that encompasses most of the probability mass. If we denote the chi-squared distribution function as $F_{\chi^2}(x)$ and a probability p, the quantile q is

$$q = F_{\chi^2}^{-1}(p). \tag{2.74}$$

From Equ. 2.74, the squared Mahalanobis distances have probability p of being less than or equal to q. When choosing p close to 1, the samples above q can be considered as outliers. The quantile being defined, we can select a weight w_q, close to 0, that we assign to samples close to q and solve Equ. 2.73 for ϵ as

$$\epsilon = \frac{1 - w_q}{w_q} \exp(-q). \tag{2.75}$$

In summary, our derivation of the Blake-Zisserman M-estimator uses two free parameters that control the boundary between inliers and outliers and the weight being assigned to outliers. We illustrate how the Blake-Zisserman M-estimator downweights outliers in Fig. 2.8.

To conclude this section, it is worth relating the M-estimator strategy to the Bayesian outlier detection algorithm proposed in [TDS07]. The approach

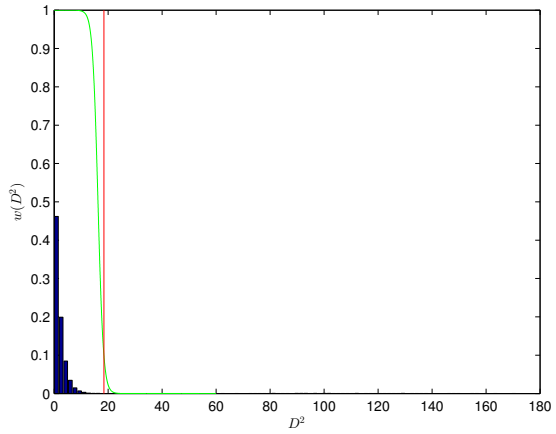


Figure 2.8: The Blake-Zisserman M-estimator naturally accounts for outliers by assigning a low weight (green curve) to data points whose squared Mahalanobis distance falls above a prescribed quantile (red line).

is outlined in the context of linear regression and can be interpreted as the Bayesian counterpart of the Blake-Zisserman M-estimator, *i.e.*, outliers are automatically downweighted in the estimation process. In contrast, the Bayesian approach is based on the EM framework, which does not have a straightforward extension to nonlinear optimization techniques, and the weights are dependent on the prediction error.

2.1.6 Regression Models

Regression analysis dates back to the nineteenth century with the seminal works of Legendre and Gauss on the *method of least-squares*. These results largely contributed to the ever unanswered question of outputting the most probable value for an unknown parameter from a set of noisy physical measurements. While Legendre notably stated that the *sample mean* minimizes the sum of the squared errors, Gauss formulated the required form of the distribution of the errors for this result to hold, giving rise to the normal distribution.

In a modern sense, regression analysis is considered as a *supervised learning* technique that aims to model the relationship between an *input* and an *output* variable. The input variable takes several names depending on the field of applications, *e.g.*, *independent variable* or *explanatory variable*. Likewise, the output variable might be referred to as a *dependent variable* or *response variable*. The input and output variables are related through a *parametric model* with unknown parameters to be estimated from the data. Although parameter learning remains the primary goal, a regression model is valuable in predicting output variables from unseen input variables. Regression analysis also deals with *non-parametric models*, *i.e.*, models described by a potentially infinite number of parameters, with methods such as *Gaussian process regression* [RW06].

In our work, we consider regression models under a slightly different perspective. In particular, we depart from the input/output variables paradigm and formulate regression in the form of the data distribution

$$\mathbf{X} \sim f_{\mathbf{X}}(\mathbf{x} \mid h(\boldsymbol{\Theta})), \tag{2.76}$$

where $\mathbf{X}$ is a measurable random variable, $\boldsymbol{\Theta}$ a latent random variable, and $h(\cdot)$ an arbitrary differentiable function such that for a given random variate $\boldsymbol{\theta}$ we have

$$\mathbb{E}[\mathbf{X}] = h(\boldsymbol{\theta}). \tag{2.77}$$

With this formulation, the goal is to estimate the unknown parameter vector $\boldsymbol{\Theta}$ using samples from $\mathbf{X}$. This can be interpreted as a generalization of the estimation framework presented thus far. Nevertheless, the classical regression can be retrieved by introducing an input variable $\mathbf{Y}$ into the function h which becomes $h(\boldsymbol{\Theta}, \mathbf{Y})$. In that case, we need to observe joint samples of $\mathbf{X}$ and $\mathbf{Y}$. For the reasons presented previously, it is common to assume a normal distribution for $f_{\mathbf{X}}$ and therefore Equ. 2.76 becomes

$$\mathbf{X} \sim \mathcal{N}(h(\boldsymbol{\theta}), \boldsymbol{\Sigma}_{\mathbf{X}}), \tag{2.78}$$

where the covariance $\boldsymbol{\Sigma}_{\mathbf{X}}$ is often regarded as a *deterministic parameter*. In a strict Bayesian formulation, $\boldsymbol{\Sigma}_{\mathbf{X}}$ should however remain a latent random variable to be estimated. When $\boldsymbol{\Theta}$ is the sole quantity of interest, $\boldsymbol{\Sigma}_{\mathbf{X}}$ is known as a *nuisance parameter* which is to be marginalized out from the posterior density [GCSR03]. If we derive an ML estimator for $\boldsymbol{\Theta}$, we obtain the same least-squares problem as in Equ. 2.65 where the squared Mahalanobis distances become

$$D_i^2 = \mathbf{e}_i^T \boldsymbol{\Sigma}_{\mathbf{X}}^{-1} \mathbf{e}_i, \tag{2.79}$$

with

$$\mathbf{e}_i = \mathbf{X}_i - h(\boldsymbol{\theta}). \tag{2.80}$$

As implied in Equ. 2.67, the minimization problem can be solved by setting the gradient of the right-hand side of Equ. 2.65 to zero, yielding the following K gradient equations for the parameter $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)^T$

$$2 \sum_{i=1}^{n} \mathbf{e}_i \frac{\partial \mathbf{e}_i}{\partial \theta_j} = 0, \quad \text{for } j = 1, \ldots, K. \tag{2.81}$$

Linear Least-Squares

If the function $h(\cdot)$ is linear in $\boldsymbol{\theta}$, the derivatives do not contain any parameters and we can rewrite Equ. 2.81 in matrix notation as

$$\mathbf{J}^T \boldsymbol{\Sigma}^{-1} \mathbf{J} \boldsymbol{\theta} = -\mathbf{J}^T \boldsymbol{\Sigma}^{-1} \mathbf{T}, \tag{2.82}$$

where $\mathbf{T} = (\mathbf{X}_1^T, \ldots, \mathbf{X}_n^T)^T$, $\mathbf{\Sigma}$ is the covariance matrix of the whole sample, and

$$\mathbf{J} = \begin{pmatrix} \frac{\partial \mathbf{e}_1^T}{\partial \theta_1} & \frac{\partial \mathbf{e}_1^T}{\partial \theta_2} & \cdots & \frac{\partial \mathbf{e}_1^T}{\partial \theta_K} \\ \frac{\partial \mathbf{e}_2^T}{\partial \theta_1} & \frac{\partial \mathbf{e}_2^T}{\partial \theta_2} & \cdots & \frac{\partial \mathbf{e}_2^T}{\partial \theta_K} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{e}_n^T}{\partial \theta_1} & \frac{\partial \mathbf{e}_n^T}{\partial \theta_2} & \cdots & \frac{\partial \mathbf{e}_n^T}{\partial \theta_K} \end{pmatrix}. \tag{2.83}$$

For the reader familiar with classical linear regression, the matrix $\mathbf{J}$ corresponds to the matrix of the negated explanatory variables. The *least-squares (LS) estimator* is therefore available in closed-form as the solution to Equ. 2.82, *i.e.*,

$$\widehat{\mathbf{\Theta}}_{LS} = -(\mathbf{J}^T \mathbf{\Sigma}^{-1} \mathbf{J})^{-1} \mathbf{J}^T \mathbf{\Sigma}^{-1} \mathbf{T}. \tag{2.84}$$

The *Fisher information matrix* derived from the total likelihood function of the sample has the analytical solution $\mathcal{I}(\boldsymbol{\theta}) = \mathbf{J}^T \mathbf{\Sigma}^{-1} \mathbf{J}$ and the variance of the estimator can be computed as

$$\mathrm{Var}\left[\widehat{\mathbf{\Theta}}_{LS}\right] = (\mathbf{J}^T \mathbf{\Sigma}^{-1} \mathbf{J})^{-1} = \mathcal{I}(\boldsymbol{\theta})^{-1}. \tag{2.85}$$

From Equ. 2.85, we can deduce that the LS estimator reaches the *Cramér-Rao bound* and can therefore be coined *efficient*. Furthermore, its sampling distribution is

$$\widehat{\mathbf{\Theta}}_{LS} \sim \mathcal{N}(\boldsymbol{\theta}, \mathcal{I}(\boldsymbol{\theta})^{-1}), \tag{2.86}$$

where $\boldsymbol{\theta}$ is the true value of the parameter. From an objective Bayesian perspective, if we set an uninformative and improper prior to $\mathbf{\Theta}$, *e.g.* $f_{\mathbf{\Theta}}(\boldsymbol{\theta}) \propto 1$, the posterior density for $\mathbf{\Theta}$ becomes

$$\mathbf{\Theta} \mid \mathbf{T} \sim \mathcal{N}(\widehat{\mathbf{\Theta}}_{LS}, \mathcal{I}(\boldsymbol{\theta})^{-1}). \tag{2.87}$$

For obvious numerical reasons, the LS estimator is rarely computed directly as in Equ. 2.84. Indeed, the matrix multiplications introduce numerous roundoff errors and the Fisher information matrix might be singular. The inverse covariance matrix $\mathbf{\Sigma}^{-1}$ is *positive semidefinite* and admits a *Cholesky decomposition* such that

$$\mathbf{\Sigma}^{-1} = \mathbf{L}\mathbf{L}^T, \tag{2.88}$$

where $\mathbf{L}$ is an invertible triangular matrix. Therefore, we can rewrite Equ. 2.82 as

$$(\mathbf{L}\mathbf{J})^T (\mathbf{L}\mathbf{J})\boldsymbol{\theta} = -(\mathbf{L}\mathbf{J})^T \mathbf{L}^{-1} \mathbf{T}, \tag{2.89}$$

which, from linear algebra, can be recognized as the *normal equations* of the overdetermined system of linear equations

$$(\mathbf{LJ})\boldsymbol{\theta} \quad = \quad -\mathbf{LT}. \tag{2.90}$$

Our least-squares estimation problem therefore involves efficient methods for solving Equ. 2.90. The various ways for solving it are well studied in numerical linear algebra and will be discussed in details in Sec. 2.2.

According to the discussions in Sec. 2.1.4, we can extend the maximum-likelihood framework by setting an informative prior on $\boldsymbol{\Theta}$ and compute an MAP estimator. We shall restrict our attention to the following type of priors

$$\boldsymbol{\Theta} \quad \sim \quad \mathcal{N}(\mathbf{0}, \lambda^{-2}\mathbf{I}), \tag{2.91}$$

where $\mathbf{0}$ is a null vector, $\mathbf{I}$ the identity matrix, and λ a real hyperparameter controlling the concentration of the prior. Plugging in the likelihood functions into Equ. 2.59, the resulting MAP estimator is the solution to the following minimization problem

$$\widehat{\boldsymbol{\Theta}}_{MAP} \quad = \quad \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left[\sum_{i=1}^{n} \mathbf{e}_i^T \boldsymbol{\Sigma}_{\mathbf{X}}^{-1} \mathbf{e}_i + \lambda^2 \boldsymbol{\theta}^T \boldsymbol{\theta} \right]. \tag{2.92}$$

In Equ. 2.92, we recognize the formulation of a *regularized least-squares problem* with the following analytical solution

$$\widehat{\boldsymbol{\Theta}}_{RLS} \quad = \quad -(\mathbf{J}^T \boldsymbol{\Sigma}^{-1} \mathbf{J} + \lambda^2 \mathbf{I})^{-1} \mathbf{J}^T \boldsymbol{\Sigma}^{-1} \mathbf{T}. \tag{2.93}$$

This form of regularization is known in the literature as *Tikhonov regularization* or *ridge regression*. The prior in Equ. 2.91 introduces a bias towards zero controlled by the parameter λ. From the large sample theory, this bias will eventually vanish as the sample size grows towards infinity. For finite samples, however, this regularization method plays an important role for solving *ill-posed* or *ill-conditioned* problems. These topics are well treated in the *inverse problem theory* and shall lead to further discussions in Sec. 2.2. In the machine learning literature, this method will rather be used as a control of *over-fitting* with the intention of performing better predictions [Bis06].

Nonlinear Least-Squares

When the function $h(\cdot)$ is nonlinear in $\boldsymbol{\theta}$, there exists no closed-form solution to the gradient equations in Equ. 2.81 and we have to resort to numerical optimization techniques. Such problems are referred to as *nonlinear least-squares (NLS) problems* and are typically solved by an iterative method that successively refines an initial guess of the parameter towards a *local minimum* [Kel99]. At each step of the algorithm, a shift vector $\Delta\boldsymbol{\theta}_{(t)}$ is computed such that the next estimate is given by

$$\widehat{\boldsymbol{\theta}}_{(t+1)} \quad \leftarrow \quad \widehat{\boldsymbol{\theta}}_{(t)} + \Delta\boldsymbol{\theta}_{(t)}. \tag{2.94}$$

The method proceeds until a predetermined set of conditions are met, *e.g*, a maximum number of iterations or a bound on the shift vector norm. This type of algorithms offers no guaranty of convergence and, if it does, the solution might not necessarily be the *global minimum* and be potentially biased. For theses reasons, the results should be cautiously interpreted.

We shall now investigate the most common method for computing the shift vector $\Delta\boldsymbol{\theta}_{(t)}$, namely the *Gauss-Newton algorithm*. The model is linearly approximated using a *Taylor series expansion* around the current estimate $\widehat{\boldsymbol{\theta}}_{(t)}$ and a local linear least-squares problem is solved. After plugging the first-order linear approximation of the model into the gradient equations in Equ. 2.81, the normal equations can be expressed as

$$\mathbf{J}_{(t)}^T\boldsymbol{\Sigma}^{-1}\mathbf{J}_{(t)}\Delta\boldsymbol{\theta}_{(t)} \quad = \quad -\mathbf{J}_{(t)}^T\boldsymbol{\Sigma}^{-1}\Delta\mathbf{T}_{(t)}, \tag{2.95}$$

where $\mathbf{J}_{(t)}$ is the *Jacobian matrix* defined in Equ. 2.83 evaluated at the current estimate $\widehat{\boldsymbol{\theta}}_{(t)}$ and $\Delta\mathbf{T}_{(t)}$ is the stacked vector of the *residuals* using $\widehat{\boldsymbol{\theta}}_{(t)}$, *i.e.*,

$$\Delta\mathbf{T}_{(t)} \quad = \quad ((\mathbf{x}_1 - h(\widehat{\boldsymbol{\theta}}_{(t)}))^T, \ldots, (\mathbf{x}_n - h(\widehat{\boldsymbol{\theta}}_{(t)}))^T)^T. \tag{2.96}$$

The assumption behind Gauss-Newton method is that the cost function can be well approximated by a quadratic function in a neighborhood around its minimum. Therefore, the updated estimate is minimizing a linear approximation of the cost function around the current estimate. For this assumption to be valid and hence ensure *local convergence*, the initial estimate needs to be close to the minimum. In contrast, the *gradient descent algorithm* is *globally convergent* in the sense that it can start from a more distant initial estimate, but at the cost of poorer convergence in the vicinity of the minimum. The Gauss-Newton algorithm can also be interpreted as an approximation to *Newton's method*, where the second-order terms of the *Hessian matrix* are ignored. Under this view, this imposes the constraint that the norm of the residuals should be small to ensure convergence.

A nonlinear least-squares method can loosely be coined an estimator, which we denote $\widehat{\boldsymbol{\Theta}}_{NLS}$, where the estimates $\widehat{\boldsymbol{\theta}}_{NLS}$ are the results of the iterative optimization. However, in contrast to the linear least-squares method, there is no direct mapping from the sample space to the estimates. In the same vein, an NLS estimator does not benefit from the attractive finite-sample properties of an LS estimator. Nevertheless, under some conditions on the nonlinear function $h(\cdot)$, the NLS estimator is consistent and asymptotically normal [Wu81] with

$$\widehat{\boldsymbol{\Theta}}_{NLS} \quad \xrightarrow{d} \quad \mathcal{N}(\boldsymbol{\theta}, \mathcal{I}^{-1}(\widehat{\boldsymbol{\theta}}_{NLS})), \tag{2.97}$$

where $\boldsymbol{\theta}$ is the true value of the sought parameter and the *observed Fisher information matrix* $\mathcal{I}(\widehat{\boldsymbol{\theta}}_{NLS}) = \mathbf{J}^T\boldsymbol{\Sigma}^{-1}\mathbf{J}$ is obtained by evaluating the Jacobian matrix $\mathbf{J}$ at the NLS estimate $\widehat{\boldsymbol{\theta}}_{NLS}$. Using a Laplace approximation [GCSR03], we can state a similar approximation in term of the posterior density of $\boldsymbol{\Theta}$, *i.e.*,

$$\boldsymbol{\Theta} \mid \mathbf{T} \quad \approx \quad \mathcal{N}(\widehat{\boldsymbol{\theta}}_{NLS}, \mathcal{I}^{-1}(\widehat{\boldsymbol{\theta}}_{NLS})). \tag{2.98}$$

As an analogy to the linear case in Equ. 2.93, we can adopt a comparable strategy for regularization and rewrite the normal equations as

$$(\mathbf{J}_{(t)}^T \boldsymbol{\Sigma}^{-1} \mathbf{J}_{(t)} + \lambda^2 \mathbf{I}) \Delta\boldsymbol{\theta}_{(t)} \quad = \quad -\mathbf{J}_{(t)}^T \boldsymbol{\Sigma}^{-1} \Delta\mathbf{T}_{(t)}, \tag{2.99}$$

The *Levenberg-Marquardt algorithm* [Mar63] is based on the update rule of Equ. 2.99, in which the λ^2 parameter is called a *damping parameter*. This latter is updated after each iteration and controls the trend of the algorithm towards a standard Gauss-Newton method (if the cost function is well approximated by its linearized version) or a pure gradient descent algorithm. Although related to Tikhonov regularization through Equ. 2.99, the Levenberg-Marquardt algorithm is not a proper tool for regularization as the damping parameter is supposed to vanish close to the minimum.

2.1.7 Identifiability and Observability

In the context of the estimation theory, we have thus far omitted the central question which pertains to assessing the *identifiability* of a statistical model. Intuitively, a parametric model is identifiable if a potentially infinite amount of its samples will uniquely determine its true parameters. Loosely speaking, we shall investigate whether we can learn something about the parameters from the available data. From a mathematical point of view, identifiability is a property of a statistical model $f(\mathbf{x};\, \boldsymbol{\theta})$, which is identifiable if two distinct values of the parameters $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ generate distinct probability density functions, *i.e.*,

$$\boldsymbol{\theta}_1 \neq \boldsymbol{\theta}_2 \quad \Longrightarrow \quad f(\mathbf{x};\, \boldsymbol{\theta}_1) \neq f(\mathbf{x};\, \boldsymbol{\theta}_2), \quad \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \Omega_{\boldsymbol{\Theta}}. \tag{2.100}$$

Whenever the model is not identifiable, we might be able to learn a subset of the parameter vector $\boldsymbol{\theta}$, in which case the model is *partially identifiable*. In some cases, it is important to restrict these definitions of identifiability to a subspace of $\Omega_{\boldsymbol{\Theta}}$ and analyze the *local identifiability* instead of the *global identifiability*. Although being a property of a model, the adjective identifiable is sometimes naturally associated to parameters. In the remainder of this thesis, we shall adopt this informal convention and freely talk about identifiable parameters.

The notion of identifiability is quite generic and we shall restrict our attention to the conditions under which a model is globally or locally identifiable. In the linear least-squares model of Sec. 2.1.6, the objective function being quadratic in $\boldsymbol{\theta}$, there exists a global minimum which is uniquely determined if the Fisher information matrix (FIM) is invertible. Therefore, a necessary and sufficient condition for global identifiability in linear regression models is the nonsingularity of the FIM. In the nonlinear case, since we are iteratively solving a linear system of equations involving the inverse of the FIM, it seems natural to extend these considerations. However, in that setting, the condition on the FIM relates to the local identifiability [Rot71] of the model at the current estimate. Since the Fisher information determines the amount of information available in a sample about an unknown parameter, analyzing its properties turns out to be a legitimate way to assess identifiability.

The concept of identifiability bears a strong resemblance with the notion of *observability* [Kal60] in *dynamical systems* or *control theory*. In those fields, a

dynamical system is usually described through a state-space representation consisting of inputs, outputs, and state variables related by differential equations. The classical definition states that a system is observable if the current state of the system can be determined only through its outputs in finite time given an initial state. If we consider the problem of *state estimation* for a discretized state-space model, *i.e.*, deriving an estimator for the internal states of the system using noisy measurements of its outputs, we can formulate it in a similar form as the regression models in Sec. 2.1.6. Furthermore, for linear dynamical systems, the identifiability condition on the FIM is equivalent to the test on the *observability matrix* [Jaz70]. For nonlinear systems, we can draw similar conclusions and relate local identifiability to *local observability* [Jau07]. Nevertheless, the definition of observability is associated with a *deterministic system*, whereas identifiability is a property of its *stochastic process* counterpart.

The whole estimation theory presented thus far can be interpreted as an instance of an *inverse problem* [Tar05]. Under that perspective, the function $h(\cdot)$ in Sec. 2.1.6 becomes the *forward operator* or *forward model*. Given fixed model parameters, the *forward problem* consists in predicting data using $h(\cdot)$. If instead of being known exactly, model parameters are represented by means of a probability density function, the forward problem is equivalent to specifying a *predictive probability density function*. Conversely, the inverse problem predicts the model parameters given fixed data using the inverse function $h^{-1}(\cdot)$, henceforth referred to as the *inverse model*. We have largely debated that the stochastic counterpart of an inverse problem boils down to determining a posterior probability density function or an estimator. As introduced by Jacques Hadamard in the early twentieth century, a problem is *well-posed* if it admits a unique solution that varies continuously with the input data, *i.e.*, small changes in the function arguments result in small changes in its outputs. In other words, the solution should be tolerant to noise in the inputs. If these conditions are not satisfied, the problem will be coined *ill-posed*. Finally, a problem with a unique solution that depends discontinuously on the input data is *ill-conditioned*. In this thesis, we shall exploit the link between estimation and inverse problems. In particular, we will borrow regularization techniques to solve ill-posed or ill-conditioned problems.

From a mathematical viewpoint, the inverse problem can be seen as an inverse function problem, *i.e.*, we seek the inverse function $h^{-1}(\cdot)$ given $h(\cdot)$. The inverse function is uniquely determined if $h(\cdot)$ is invertible. For $h(\cdot)$ to be invertible, it needs to be *injective*, *i.e.*, each element of its *domain* maps to a unique element of its *codomain*. This interpretation sheds a new light on the observability concept. If the forward model $h(\cdot)$ is not injective, its parameter is not observable. If the function $h(\cdot)$ is injective on a subset of its domain, the parameter is locally observable. If the domain of $h(\cdot)$ consists in a set of vectors and the function becomes injective by reducing the dimensionality of the domain, the parameter is *partially observable*.

From a pure Bayesian perspective, identifiability may be regarded as an artificial concept [XC06]. Indeed, by setting a proper prior, we can always estimate the parameter. We have already encountered such an example in Sec. 2.1.6 and shown that it relates to regularization. However, with the advent of uninformative priors, the problem resurrects and one might be interested in what has been learned from the data so far and what can potentially be learned with an infinite amount of data.

2.2 Linear Algebra

Linear algebra is a branch of mathematics that is concerned with the study of linear mappings between vector spaces and originates from the attempts to solve systems of linear equations in the seventeenth century. As probability theory, linear algebra is a purely theoretical science that plays a fundamental role in a variety of applied sciences such as statistics, econometrics, or engineering. After a rather informal introduction to the main theoretical concepts, the rest of this section will be essentially dedicated to the application of linear algebra theory for solving systems of linear equations in the form given by the least-squares problems in Sec. 2.1.6. We refer the interested reader to the numerous textbooks on the topic for a comprehensive overview, *e.g.* [GL96] and the references therein.

2.2.1 Basic Definitions

The primary object of interest in linear algebra is a *vector space*. A vector space $\mathcal{V}$ is a collection of elements, called *vectors*, which admit a set of operations satisfying certain requirements. Vectors spaces are defined over *fields* $\mathcal{F}$ whose elements are called *scalars*. In the following, we will adopt the standard boldface notation for vectors (*e.g.* $\mathbf{v}$), and the light notation for scalars (*e.g.* c). If a subset $\mathcal{W}$ of $\mathcal{V}$ is also a vector space, it is referred to as a *subspace*. The set of all possible linear combinations of a set of vectors $\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ from $\mathcal{V}$ forms a subspace called the *span*, *i.e.*,

$$\text{span}\left[\{\mathbf{v}_1, \dots, \mathbf{v}_n\}\right] = \left\{\sum_{i=1}^{n} c_i \mathbf{v}_i \mid c_i \in \mathcal{F}\right\}. \tag{2.101}$$

Furthermore, if the only way to generate the *null vector* is to set all the scalars c_i to zero, the set of vectors is said to be *linearly independent* and its elements form a *basis* for its span. The *dimension* of a vector space is the cardinality of one of its bases, each of them having the same number of elements. An equivalent definition for linear independence is that none of the vectors in the set can be expressed as a linear combination of the others. As a consequence thereof, every basis in a vector space establishes a *coordinate system*, *i.e.*, each vector is uniquely determined by the scalars, called *coordinates*, in the linear combination.

A *linear map* is a mapping f that transforms vectors from $\mathcal{V}$ into $\mathcal{W}$. The *range* or *image* of the linear map f is the set defined by

$$\text{range}\left[f\right] = \{\mathbf{w} \in \mathcal{W} \mid \mathbf{w} = f(\mathbf{v}), \mathbf{v} \in \mathcal{V}\}. \tag{2.102}$$

In the same direction, the *kernel* or *nullspace* of the linear map f is defined by

$$\text{kernel}\left[f\right] = \{\mathbf{v} \in \mathcal{V} \mid f(\mathbf{v}) = \mathbf{0}, \mathbf{0} \in \mathcal{W}\}. \tag{2.103}$$

The range is a subspace of $\mathcal{W}$ and the kernel a subspace of $\mathcal{V}$. Furthermore, the *rank-nullity theorem* states that

$$\dim\left[\text{kernel}\left[f\right]\right] + \dim\left[\text{range}\left[f\right]\right] \quad = \quad \dim\left[\mathcal{V}\right]. \tag{2.104}$$

The dimension of the range is called the *rank* of f and the dimension of the kernel its *nullity*.

Every linear map f from $\mathcal{V}$ to $\mathcal{W}$ can be represented as a *matrix*. If we define a basis for $\mathcal{V}$ as $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ and for $\mathcal{W}$ as $\{\mathbf{w}_1, \ldots, \mathbf{w}_m\}$, the linear map is entirely described by an $m \times n$ matrix $\mathbf{A}$. That is, we can express any vector $\mathbf{v} \in \mathcal{V}$ as an $n \times 1$ matrix, or column vector, and any vector $\mathbf{w} \in \mathcal{W}$ as an $m \times 1$ matrix, and the range of the linear map becomes

$$\text{range}\left[\mathbf{A}\right] \quad = \quad \left\{\mathbf{w} \in \mathcal{W} \mid \mathbf{w} = \mathbf{A}\mathbf{v}, \mathbf{v} \in \mathcal{V}\right\}. \tag{2.105}$$

A similar consideration allows us to express the nullspace as

$$\text{null}\left[\mathbf{A}\right] \quad = \quad \left\{\mathbf{v} \in \mathcal{V} \mid \mathbf{A}\mathbf{v} = \mathbf{0}, \mathbf{0} \in \mathcal{W}\right\}, \tag{2.106}$$

where we have used the more standard notation for the nullspace of a matrix. If we now consider the set $\{\mathbf{a}_1, \ldots, \mathbf{a}_n\}$ of column vectors of $\mathbf{A}$, the range can also be defined as

$$\text{range}\left[\mathbf{A}\right] \quad = \quad \text{span}\left[\{\mathbf{a}_1, \ldots, \mathbf{a}_n\}\right], \tag{2.107}$$

which is also known as the *column space* of $\mathbf{A}$. We shall further introduce the rank operator for a matrix as

$$\text{rank}\left[\mathbf{A}\right] \quad = \quad \dim\left[\text{range}\left[\mathbf{A}\right]\right]. \tag{2.108}$$

Intuitively, the rank cannot be bigger than either m or n, and if it is equal to $\min(m, n)$, the matrix is said to have *full rank*. Otherwise, it will be *rank deficient*. An alternative condition for the matrix having full rank is that its column vectors form a basis. In this thesis, we will restrict our attention to the field $\mathbb{R}$ of real numbers and denote a vector space of dimension m as $\mathbb{R}^m$. Likewise, we can define a vector space on matrices with dimension $m \times n$, which we will denote $\mathbb{R}^{m \times n}$. In the real field, an important property of the rank is that

$$\text{rank}\left[\mathbf{A}\right] \quad = \quad \text{rank}\left[\mathbf{A}^T\right] \quad = \quad \text{rank}\left[\mathbf{A}\mathbf{A}^T\right] \quad = \quad \text{rank}\left[\mathbf{A}^T\mathbf{A}\right]. \tag{2.109}$$

Lastly, Equ. 2.104 applies to the matrix representation as follows

$$\dim\left[\text{null}\left[\mathbf{A}\right]\right] + \text{rank}\left[\mathbf{A}\right] \quad = \quad n. \tag{2.110}$$

2.2.2 Matrix Analysis

The concept of rank as defined in Sec. 2.2.1 is central to solving systems of linear equations and, in general, to many algorithms in linear algebra. Therefore, we shall investigate efficient methods for accurately determining the rank of a matrix. The implementation of those algorithms on a computer introduces numerous roundoff errors and we shall thus account for numerical issues. Apart from the floating point operations, these aspects also appear when the matrices are constructed from noisy data.

As a motivating example, we shall consider the following matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}. \tag{2.111}$$

By manual screening or reduction to its *row echelon form*, it is straightforward to conclude that matrix $\mathbf{A}$ is rank deficient with rank 2 (the second column is a linear combination of the others). If we now corrupt $\mathbf{A}$ with random noise, we end up with the following matrix

$$\mathbf{A}^* = \begin{pmatrix} 1.0003 & 1.9983 & 3.0003 \\ 3.9992 & 4.9999 & 6.0003 \\ 7.0014 & 7.9998 & 8.9991 \end{pmatrix}. \tag{2.112}$$

From an algebraic perspective, the matrix $\mathbf{A}^*$ has strictly full rank. However, as it has been constructed from a rank-deficient matrix with random perturbations, it will cause issues if treated as a full-rank matrix as we shall witness later in this section. It is highly probable that most of the matrices in real-world applications will resemble $\mathbf{A}^*$ and appear to have full algebraic rank. This illustrative example is leading us towards methods for reliable rank detection in presence of noise. In the rest of this subsection, we will thus briefly present two *rank-revealing decompositions* that can cope with these numerical difficulties.

Singular Value Decomposition

The *singular value decomposition (SVD)* is the most powerful tool for determining the rank of a matrix associated with numerical errors. For a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, there exists a factorization of the form

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \tag{2.113}$$

where $\mathbf{U} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{n \times n}$ are *orthogonal matrices*, and $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ is a *diagonal matrix* of the form

$$\mathbf{\Sigma} = \operatorname{diag}\left[\sigma_1, \ldots, \sigma_p\right], \quad p = \min(m, n). \tag{2.114}$$

The values σ_i are called the *singular values* of $\mathbf{A}$, the m column vectors of $\mathbf{U}$ the *left-singular vectors* of $\mathbf{A}$, and the n column vectors of $\mathbf{V}$ the *right-singular vectors* of $\mathbf{A}$. The singular values are nonnegative and listed in descending

order. We denote the maximum singular value of $\mathbf{A}$ as $\sigma_{max}(\mathbf{A})$, the minimum as $\sigma_{min}(\mathbf{A})$, and, depending on the context, the i-th singular value as $\sigma_i(\mathbf{A})$.

The SVD reveals much information about the structure of a matrix. The algebraic rank, as defined in Sec. 2.2.1, is the number of nonzero singular values, *i.e.*,

$$\text{rank}\,[\mathbf{A}] \quad = \quad r, \quad r = \text{argmax}_i\, \sigma_i \neq 0. \tag{2.115}$$

If we further denote the set of column vectors of $\mathbf{U}$ as $\{\mathbf{u}_1, \ldots, \mathbf{u}_m\}$, the range of $\mathbf{A}$ is

$$\text{range}\,[\mathbf{A}] \quad = \quad \text{span}\,[\{\mathbf{u}_1, \ldots, \mathbf{u}_r\}]\,. \tag{2.116}$$

If we denote the set of column vectors of $\mathbf{V}$ as $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, the nullspace of $\mathbf{A}$ is

$$\text{null}\,[\mathbf{A}] \quad = \quad \text{span}\,[\{\mathbf{v}_{r+1}, \ldots, \mathbf{v}_n\}]\,. \tag{2.117}$$

Finally, the matrix $\mathbf{A}$ can be expressed as an *SVD expansion* of the form

$$\mathbf{A} \quad = \quad \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T. \tag{2.118}$$

In practice, a lightweight version of the SVD, called *thin SVD*, is used to equivalently factorize matrices. In the common case where $m \geq n$, we need at most n columns of $\mathbf{U}$, therefore the thin SVD is

$$\mathbf{A} \quad = \quad \mathbf{U}_n \mathbf{\Sigma}_n \mathbf{V}^T, \tag{2.119}$$

where $\mathbf{U}_n \in \mathbb{R}^{m \times n}$ and $\mathbf{\Sigma}_n \in \mathbb{R}^{n \times n}$. Furthermore, if we examine Equ. 2.118, we see that actually only r vectors are needed to fully represent the matrix. Therefore, the *compact SVD* factorizes $\mathbf{A}$ as

$$\mathbf{A} \quad = \quad \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^T, \tag{2.120}$$

where $\mathbf{U}_r \in \mathbb{R}^{m \times r}$, $\mathbf{\Sigma}_r \in \mathbb{R}^{r \times r}$, and $\mathbf{V}_r \in \mathbb{R}^{n \times r}$.

Thus far, we have seen that SVD reveals the algebraic rank of a matrix as the number of its nonzero singular values. To cope with near rank-deficient matrices such as in Equ. 2.112, we need to refine the definitions by first introducing the concept of *matrix norm*. The matrix *p-norm* is an extension of the vector norm and is generally defined as

$$\|\mathbf{A}\|_p \quad = \quad \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{A}\mathbf{x}\|_p}{\|\mathbf{x}\|_p}, \quad p \geq 1. \tag{2.121}$$

For arbitrary p, Equ. 2.121 requires an optimization method. In the case of $p = 1$ and $p = \infty$, we have the following analytical solutions

$$\|\mathbf{A}\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{m} |a_{ij}|\,, \tag{2.122}$$

$$\|\mathbf{A}\|_\infty = \max_{1\leq i\leq m} \sum_{j=1}^{n} |a_{ij}|\,, \tag{2.123}$$

where a_{ij} are the coefficients of the matrix $\mathbf{A}$. In other words, the 1-norm is the maximum absolute column sum and the ∞-norm the maximum absolute row sum. Another important matrix norm is the so-called *Frobenius norm* defined as

$$\|\mathbf{A}\|_F = \sqrt{\mathrm{tr}\left(\mathbf{A}^T\mathbf{A}\right)}, \tag{2.124}$$

which consists in the square root of the sum of the diagonal elements of $\mathbf{A}^T\mathbf{A}$. Finally, we can relate the Frobenius norm and the 2-norm to the singular values of $\mathbf{A}$ as follows

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{r} \sigma_i^2}, \tag{2.125}$$

$$\|\mathbf{A}\|_2 = \sigma_{max}(\mathbf{A}). \tag{2.126}$$

Unlike the vector norm, the notion of matrix norm is relatively hard to comprehend. Nevertheless, we can interpret it in similar way by thinking of a distance or magnitude in the vector space of matrices.

Equipped with the matrix norm concept, we can now formally define the *numerical rank* [GL96] r_ϵ of a matrix $\mathbf{A}$ as

$$\mathrm{rank}\,[\mathbf{A}, \epsilon] = \min_{\|\mathbf{A}-\mathbf{B}\|_2\leq\epsilon} \mathrm{rank}\,[\mathbf{B}]\,. \tag{2.127}$$

Intuitively, this definition states that the numerical rank of $\mathbf{A}$ is the minimum algebraic rank over all matrices $\mathbf{B}$ in a neighborhood of $\mathbf{A}$ bounded by ϵ. Applying this definition to the examples in Equ. 2.111 and Equ. 2.112, we see that, for a suitably chosen ϵ, the matrix $\mathbf{A}$ and $\mathbf{A}^*$ are in the same neighborhood. Therefore, the numerical rank of $\mathbf{A}^*$ is the algebraic rank of $\mathbf{A}$, *i.e.* 2. Continuing on this example, if we now assume that $\mathbf{A}^* = \mathbf{A} + \mathbf{E}$, where $\mathbf{E}$ is a perturbation matrix, the distance between $\mathbf{A}$ and $\mathbf{A}^*$ is $\|\mathbf{E}\|_2$. It follows that if we want to retrieve the matrix $\mathbf{A}$ from $\mathbf{A}^*$, we should define a search region around $\mathbf{A}^*$ of a least $\|\mathbf{E}\|_2$ and therefore ideally set ϵ to $\|\mathbf{E}\|_2$. In a more general sense, as long as the magnitude of the perturbation matrix is bounded by ϵ, the numerical rank of the perturbed matrix corresponds to the algebraic rank of the original matrix. In practice, $\mathbf{E}$ is not known and its magnitude thus needs to be estimated. If we have a statistical model for the source of errors, we can derive an estimator for $\|\mathbf{E}\|_2$ [Han88]. Otherwise, we have to rely on a heuristic.

In any case, even if the matrix is given exactly, floating point operations will always introduce an error which is proportional to the machine epsilon.

In order to come up with an operational definition of the numerical rank, we shall first introduce the *low-rank approximation* $\mathbf{A}_k$ of a rank-r matrix $\mathbf{A}$ with SVD given by Equ. 2.120 as

$$\mathbf{A}_k \quad = \quad \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T, \quad k < r, \tag{2.128}$$

i.e., we approximate $\mathbf{A}$ using only k singular vectors and values. Given the approximation in Equ. 2.128, it can be demonstrated [GL96] that

$$\|\mathbf{A} - \mathbf{A}_k\|_2 \quad = \quad \sigma_{k+1} \tag{2.129}$$

and that $\mathbf{A}_k$ is the closest rank-k matrix to $\mathbf{A}$. In other words, if $\mathbf{A}$ has full rank, its smallest singular value corresponds to the distance in matrix space to the closest rank-deficient matrix. Consequently, a full-rank matrix with small singular values is nearly rank deficient.

Using Equ. 2.127 and Equ. 2.129, the numerical rank r_ϵ of $\mathbf{A}$ can be related to its singular values, which satisfy the inequality

$$\sigma_{r_\epsilon} \quad > \quad \epsilon \quad \geq \quad \sigma_{r_\epsilon+1}. \tag{2.130}$$

Furthermore, for the numerical rank r_ϵ to make any sense, it should be robust to small perturbations in ϵ, *i.e.*, there should be a well-determined gap between σ_{r_ϵ} and $\sigma_{r_\epsilon+1}$. For this reason, if the singular values gradually decay to zero without any significant gap, the numerical rank is ill-defined. In [GKS76], the authors circumvent this issue by augmenting Equ. 2.130 with an additional parameter δ controlling the distance between ϵ and σ_{r_ϵ} and define the numerical rank with the triplet (r, ϵ, δ).

Along with the numerical rank, we may also define the *numerical range* and the *numerical nullspace* by

$$\text{range}\left[\mathbf{A}, \epsilon\right] \quad = \quad \text{span}\left[\{\mathbf{u}_1, \ldots, \mathbf{u}_{r_\epsilon}\}\right], \tag{2.131}$$

$$\text{null}\left[\mathbf{A}, \epsilon\right] \quad = \quad \text{span}\left[\{\mathbf{v}_{r_\epsilon+1}, \ldots, \mathbf{v}_n\}\right]. \tag{2.132}$$

As pointed out by Hansen in [Han98], one should assume that the rows or the columns of the matrix have the same magnitude in order to interpret Equ. 2.130 correctly, *i.e.*, the errors on the coefficients of the matrix should have the same scale. In this work dealing with parameters at various scales, we choose to equilibrate the columns so that they have the same norm. This can be done by right-multiplying the matrix $\mathbf{A}$ by a scaling matrix $\mathbf{G}$ defined as

$$\mathbf{G} \quad = \quad \text{diag}\left[\frac{1}{\|\mathbf{a}_1\|_2}, \ldots, \frac{1}{\|\mathbf{a}_n\|_2}\right], \tag{2.133}$$

where the $\mathbf{a}_i$ are the column vectors of $\mathbf{A}$. Furthermore, it has been shown in [vdS69] that this choice of $\mathbf{G}$ minimizes the *condition number* of $\mathbf{AG}$. When

solving a system of linear equations, the condition number is a measure of the *sensitivity* of the solution. A large condition number means that the solution is not stable when applying small perturbations to the system. Such problems are referred to as *ill-conditioned problems*. In statistical terms, an unstable solution will correspond to a large variance of the estimator.

In regards to the concerns raised in Sec. 2.1.7, the SVD readily provides answers to the questions of identifiability and observability. We have seen that both concepts require that the Fisher information matrix is nonsingular, which is equivalent to it having full rank. Along the same lines, we can express these concepts in term of the numerical rank. We shall thus state that a model is *numerically observable* or *numerically identifiable* if the associated Fisher information matrix has full numerical rank. Furthermore, the numerical nullspace will correspond to directions in the parameter space that are numerically unobservable or unidentifiable.

QR Decomposition

As demonstrated above, the singular value decomposition is the prevailing technique to analyze a matrix and identify numerical rank deficiency. For large matrices, unfortunately, it becomes computationally involved and thus time and memory consuming. For this reason, it is advantageous to consider an alternative matrix decomposition. The *QR decomposition* (named after conventional matrix notation) of a full-rank matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $m \geq n$ takes the form

$$\mathbf{A} \quad = \quad \begin{pmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{pmatrix} \begin{pmatrix} \mathbf{R}_1 \\ \mathbf{0} \end{pmatrix}, \tag{2.134}$$

where $\mathbf{Q}_1 \in \mathbb{R}^{m \times n}$ and $\mathbf{Q}_2 \in \mathbb{R}^{m \times (m-n)}$ have orthonormal columns, and $\mathbf{R}_1 \in \mathbb{R}^{n \times n}$ is an *upper-triangular matrix*. As for SVD, we can express a lightweight decomposition, called *thin QR*, as

$$\mathbf{A} \quad = \quad \mathbf{Q}_1 \mathbf{R}_1. \tag{2.135}$$

For full-rank matrices, the thin QR decomposition is unique. Furthermore, the columns of $\mathbf{Q}_1$ form an *orthonormal basis* for the range of $\mathbf{A}$, the columns of $\mathbf{Q}_2$ an orthonormal basis for the nullspace of $\mathbf{A}^T$ or for the *orthogonal complement* of the range of $\mathbf{A}$, and $\mathbf{R}_1$ is the *Cholesky factor* of $\mathbf{A}\mathbf{A}^T$.

In that form, this decomposition is not particularly useful for rank-deficient matrices. Fortunately, a column-pivoting strategy allows us to produce a *rank-revealing QR (RRQR) decomposition* [Cha87] in the form given by

$$\mathbf{A}\mathbf{\Pi} \quad = \quad \begin{pmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{pmatrix} \begin{pmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} \\ \mathbf{0} & \mathbf{R}_{22} \end{pmatrix}, \tag{2.136}$$

where $\mathbf{\Pi} \in \mathbb{R}^{n \times n}$ is a *permutation matrix* whose goal is to move the most linearly independent columns of $\mathbf{A}$ to the front, $\mathbf{R}_{11} \in \mathbb{R}^{r_\epsilon \times r_\epsilon}$ is an upper-triangular matrix, $\mathbf{R}_{12} \in \mathbb{R}^{r_\epsilon \times (n-r_\epsilon)}$, $\mathbf{R}_{22} \in \mathbb{R}^{(m-r_\epsilon) \times (n-r_\epsilon)}$, and r_ϵ is the numerical rank of $\mathbf{A}$. In analogy with Equ. 2.130, an RRQR decomposition satisfies the following inequality

$$\sigma_{r_\epsilon}(\mathbf{A}) \geq \sigma_{min}(\mathbf{R}_{11}) > \epsilon \geq \|\mathbf{R}_{22}\|_2 \geq \sigma_{r_\epsilon+1}(\mathbf{A}). \tag{2.137}$$

Here, the criterion on the well-determined gap between σ_{r_ϵ} and $\sigma_{r_{\epsilon+1}}$ applies on the distance between $\sigma_{min}(\mathbf{R}_{11})$ and $\|\mathbf{R}_{22}\|_2$. Although we will not delve into the specific implementation details of the algorithms producing such a decomposition, we shall state some of its properties. First, the columns of $\mathbf{Q}_1 \in \mathbb{R}^{m\times r_\epsilon}$ form an orthonormal basis for the numerical range of $\mathbf{A}$ and the columns of $\mathbf{Q}_2 \in \mathbb{R}^{m\times(m-r_\epsilon)}$ an orthonormal basis for the numerical nullspace of $\mathbf{A}^T$ or for the orthogonal complement of the numerical range of $\mathbf{A}$. Then, if the numerical rank is equal to the algebraic rank of $\mathbf{A}$, *i.e.* $r = r_\epsilon$, the matrix $\mathbf{R}_{22}$ is null. Finally, it is worth noting that an RRQR decomposition is not as reliable as an SVD for determining the numerical rank of a matrix [CH92].

2.2.3 Solutions to Least-Squares Problems

One major application of linear algebra is the calculation of solutions to overdetermined systems of linear equations of the form

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{2.138}$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a *data matrix*, $\mathbf{b} \in \mathbb{R}^m$ an *observation vector*, $\mathbf{x} \in \mathbb{R}^n$ a *parameter vector*, and $m > n$. Such systems usually have no exact solution and a possible approximate solution is sought by minimizing the 2-norm of the residual vector, which yields the least-squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 . \tag{2.139}$$

Some linear algebra arguments [GL96] show that the solution $\mathbf{x}_{LS}$ to this problem satisfies the *normal equations*

$$\mathbf{A}^T\mathbf{A}\mathbf{x}_{LS} = \mathbf{A}^T\mathbf{b}, \tag{2.140}$$

which can be solved directly as

$$\mathbf{x}_{LS} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}. \tag{2.141}$$

At the least-squares solution, the residual vector

$$\mathbf{r}_{LS} = \mathbf{b} - \mathbf{A}\mathbf{x}_{LS} \tag{2.142}$$

is orthogonal to the range of $\mathbf{A}$, which reveals the etymology of the term "normal equations". In Sec. 2.1.6, we have also noticed that the maximum-likelihood estimator of a linear regression model with normally distributed observations corresponds to Equ. 2.141. This relation is a plausible explanation for the origin of the term "normal distribution". In a more general sense, maximum-likelihood

estimators for nonlinear regression models involve at each iteration a problem of the form given by Equ. 2.140. The tight interplay between statistical estimation and linear algebra will thus be beneficial for solving our estimation problems. In particular, we can employ the matrix decompositions of Sec. 2.2.2 to work on Equ. 2.138 rather than purely inverting $\mathbf{A}^T\mathbf{A}$ as in Equ. 2.141.

From the linear algebra theory, we know that Equ. 2.140 has a unique solution if and only if $\mathbf{A}^T\mathbf{A}$ is invertible, *i.e.*, it has full rank. Then, from Equ. 2.109, we can conclude that this unique solution only exists if $\mathbf{A}$ has full rank. If $\mathbf{A}$ is rank deficient, there is an infinite number of solutions that minimize the 2-norm of the residual vector. For instance, if we denote one of these solutions by $\mathbf{x}_0$ and a vector in the nullspace of $\mathbf{A}$ by $\mathbf{z}$, then

$$\|\mathbf{A}(\mathbf{x}_0 + \mathbf{z}) - \mathbf{b}\|_2 \quad = \quad \|\mathbf{A}\mathbf{x}_0 + \mathbf{A}\mathbf{z} - \mathbf{b}\|_2 \quad = \quad \|\mathbf{A}\mathbf{x}_0 - \mathbf{b}\|_2 \,. \tag{2.143}$$

Therefore, $\mathbf{x}_0 + \mathbf{z}$ also minimizes the 2-norm of the residual.

In the remainder of this section, we shall investigate how the matrix decompositions presented thus far can readily provide solutions to the least-squares problem. We will start with the case of $\mathbf{A}$ having full rank and continue the discussion on methods for solving rank-deficient problems.

Full-Rank Problems

The least-squares solution to Equ. 2.138 is given by

$$\mathbf{x}_{LS} \quad = \quad \mathbf{A}^{\dagger}\mathbf{b}, \tag{2.144}$$

where $\mathbf{A}^{\dagger}$ is the *pseudoinverse* [CM09] of $\mathbf{A}$ that can be expressed in term of its SVD as

$$\mathbf{A}^{\dagger} \quad = \quad \sum_{i=1}^{n} \frac{\mathbf{v}_i \mathbf{u}_i^T}{\sigma_i}. \tag{2.145}$$

Therefore, the least-squares solution is directly available from the SVD of $\mathbf{A}$ as

$$\mathbf{x}_{LS} \quad = \quad \sum_{i=1}^{n} \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i. \tag{2.146}$$

From Equ. 2.146, it is worth noting that if $\mathbf{A}$ is nearly rank deficient, its smallest singular value will just amplify the noise in $\mathbf{b}$ due to the division and hence the variability of $\mathbf{x}_{LS}$. This fact is even more apparent if we compute the covariance matrix of $\mathbf{x}_{LS}$ in term of the SVD, given by

$$\mathrm{Var}\left[\mathbf{x}_{LS}\right] \quad = \quad \sum_{i=1}^{n} \frac{\mathbf{v}_i \mathbf{v}_i^T}{\sigma_i^2}. \tag{2.147}$$

The matrix pseudoinverse can also be formulated in term of the QR decomposition of $\mathbf{A}$ from Equ. 2.134 as

$$\mathbf{A}^{\dagger} = \mathbf{R}_1^{-1}\mathbf{Q}_1^T, \tag{2.148}$$

which yields the least-squares solution

$$\mathbf{x}_{LS} = \mathbf{R}_1^{-1}\mathbf{Q}_1^T\mathbf{b}. \tag{2.149}$$

The matrix $\mathbf{R}_1$ being upper triangular, Equ. 2.149 can be solved by *back substitution* instead of computing an inverse. Furthermore, $\mathbf{Q}_1^T\mathbf{b}$ can be efficiently computed if $\mathbf{Q}_1$ is stored as *Householder vectors*. In terms of the QR decomposition, the covariance matrix is expressed as

$$\mathrm{Var}\left[\mathbf{x}_{LS}\right] = (\mathbf{R}_1^T\mathbf{R}_1)^{-1}, \tag{2.150}$$

which can be calculated at low cost when using the triangular form of $\mathbf{R}_1$ [QSS07].

Rank-Deficient Problems and Regularization

As demonstrated above, the rank-deficient least-squares problem has an infinite number of solutions. If $\mathrm{rank}\left[\mathbf{A}\right] = r < n$, the pseudoinverse solution in Equ. 2.146 can be written as

$$\mathbf{x}_{LS} = \sum_{i=1}^{r} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i. \tag{2.151}$$

Furthermore, this solution has minimum 2-norm amongst all the minimizers of Equ. 2.139 [GL96].

If we recall the considerations of Sec. 2.2.2, a matrix will rarely be algebraically rank deficient. Instead, it will have small singular values that dominate the pseudoinverse solution. As a consequence, although minimizing the 2-norm of the residual, the pseudoinverse solution will have large magnitude and variance. In fact, the solution is just over-fitting the data and will poorly generalize. In the nonlinear least-squares problem, the large magnitude of the solution introduces large jumps between iterations and the updated estimate might fall in a region where the model is not well approximated by a linear function, eventually causing the divergence of the algorithm. From the perspective of statistical estimation, the least-squares solution is the one with lowest variance amongst all unbiased estimators (MVU). Nevertheless, we might produce an estimator with much lower mean-squared error (MMSE) by introducing a bias which reduces the variance (*cf.* Equ. 2.48). This goal can be achieved by means of *regularization techniques*.

The most common method for regularization is to transform the original problem in Equ. 2.139 into

$$\min_{\mathbf{x}\in\mathbb{R}^n} \left\|\mathbf{A}_k\mathbf{x} - \mathbf{b}\right\|_2, \tag{2.152}$$

where $\mathbf{A}_k$ is the low-rank approximation of $\mathbf{A}$ given by Equ. 2.128. The minimum 2-norm solution to Equ. 2.152 is given by

$$\mathbf{x}_{TSVD} = \mathbf{A}_k^{\dagger}\mathbf{b} = \sum_{i=1}^{k} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i. \tag{2.153}$$

This method dates back from the seventies and has been coined *truncated SVD (TSVD)* [Han87]. It is closely related to dimensionality reduction in *principal component analysis (PCA)* or noise filtering in *fast Fourier transform (FFT)*. Furthermore, TSVD can be interpreted as a special case of the Tikhonov regularization presented in Sec. 2.1.6 [Han87]. Indeed, the TSVD solution can be expressed as

$$\mathbf{x}_{TSVD} = \sum_{i=1}^{r} \frac{\mathbf{u}_i^T\mathbf{b}f(\sigma_i)}{\sigma_i}\mathbf{v}_i, \tag{2.154}$$

where the filter function f is defined by

$$f(\sigma) = \begin{cases} 1 & \text{if } \sigma \geq \sigma_k \\ 0 & \text{otherwise,} \end{cases} \tag{2.155}$$

whereas for Tikhonov regularization, the filter function is

$$f(\sigma) = \frac{\sigma^2}{\sigma^2 + \lambda^2}. \tag{2.156}$$

Therefore, while Tikhonov regularization smoothly dampens the small singular values compared to λ, TSVD acts as a sharp filter, as can be seen in Fig. 2.9. In both cases, the parameter k or λ controls the trade-off between bias and variance of the estimator. With respect to the discussions in Sec. 2.2.2, the algorithmic parameter k for TSVD should correspond to the numerical rank r_ϵ of the matrix $\mathbf{A}$. Indeed, any singular value of the perturbed matrix smaller than the magnitude of the noise cannot be distinguished from a true zero singular value of the original matrix.

From the RRQR decomposition of $\mathbf{A}$, the minimum 2-norm least-squares solution can unfortunately not be directly computed. If $\text{rank}\,[\mathbf{A}] = r$, the matrix $\mathbf{R}_{22}$ in Equ. 2.136 is null. The remaining matrices $\mathbf{R}_{11}$ and $\mathbf{R}_{12}$ can be further factorized as follows

$$\mathbf{Z}_1^T \begin{pmatrix} \mathbf{R}_{11}^T \\ \mathbf{R}_{12}^T \end{pmatrix} = \mathbf{T}_{11}^T, \tag{2.157}$$

where $\mathbf{Z}_1 \in \mathbb{R}^{n\times r}$ corresponds to the first r columns of $\mathbf{Z} \in \mathbb{R}^{n\times n}$, the $\mathbf{Q}$ matrix in the QR decomposition of $(\mathbf{R}_{11}^T, \mathbf{R}_{12}^T)^T$, and $\mathbf{T}_{11} \in \mathbb{R}^{r\times r}$ is a *lower-triangular matrix*. This type of decomposition is referred to as a *complete orthogonal decomposition (COD)* [GL96] and the minimum 2-norm least-squares solution is given by

$$\mathbf{x}_{LS} = \mathbf{\Pi}\mathbf{Z} \begin{pmatrix} \mathbf{T}_{11}^{-1}\mathbf{Q}_1^T\mathbf{b} \\ \mathbf{0} \end{pmatrix}. \tag{2.158}$$

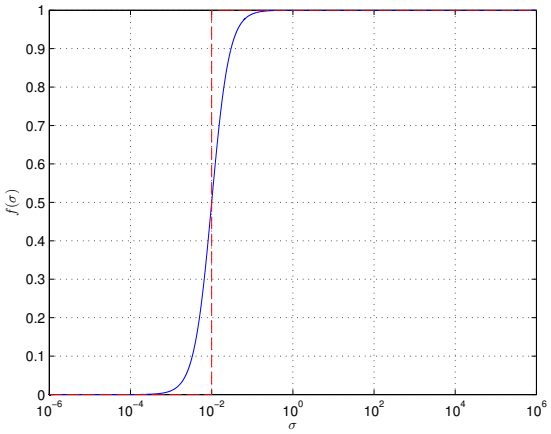


Figure 2.9: Comparison of the filter functions for TSVD (red) and Tikhonov (blue) regularization. The sharp filter of the TSVD method can be interpreted as an approximation to the smooth filter of the Tikhonov regularization method.

If the matrix $\mathbf{A}$ is numerically rank deficient, the matrix $\mathbf{R}_{22}$ is not null, but small in norm. Therefore, we can follow the same strategy as for the TSVD, *i.e.*, approximating $\mathbf{A}$ with a lower-rank matrix $\mathbf{A}_{r_\epsilon}$ which is formed by dropping $\mathbf{R}_{22}$ in the RRQR decomposition. We can then find the least-squares solution of the regularized problem with the COD of $\mathbf{A}_{r_\epsilon}$ as in Equ. 2.158. The resulting *truncated QR (TQR)* solution is an approximation to the TSVD solution.

In some applications, an over-parameterized model can lead to a data matrix which remains rank deficient even with an infinite amount of data. An over-parameterized model might arise in practice when there is no analytical way to simplify it beforehand [IKP11]. In that case, it makes sense to predict the observations with a restricted set of parameters. The optimal selection of these parameters is known as *subset selection* [GL96]. In the RRQR decomposition, the permutation matrix $\mathbf{\Pi}$ moves the most linearly independent columns of $\mathbf{A}$ to the front of $\mathbf{A\Pi}$. Therefore, if we pick the first r columns of $\mathbf{A\Pi}$, we can solve a full-rank least-squares problem while leaving the $n - r$ remaining parameters to zero. This solution is referred to as the *basic solution* and is computed as

$$\mathbf{x}_B = \mathbf{\Pi} \begin{pmatrix} \mathbf{R}_{11}^{-1}\mathbf{Q}_1^T\mathbf{b} \\ \mathbf{0} \end{pmatrix}. \tag{2.159}$$

In the case of numerical rank deficiency, r can be replaced by an estimate of the numerical rank r_ϵ. Although the norm of $\mathbf{x}_B$ is in general larger than $\mathbf{x}_{LS}$, it comes at lower computational cost. When to use one or the other is dependent on the application.

3 Online Self-Calibration Algorithm

Inductive inference is the only process known to us by which essentially new knowledge comes into the world.

Sir Ronald Aymler Fisher (1890-1962)

Now that we have exposed the principal mathematical instruments and a rigorous formalism, we shall present the core theoretical and practical contributions of this thesis. We propose a generic algorithm for the extrinsic and/or intrinsic calibration of robotic sensors based on a formulation of their probabilistic data model. The extrinsic calibration of a sensor consists in retrieving the rigid transformation between its *local coordinate frame* and a *reference coordinate frame*, *e.g.*, a *world coordinate frame* or *vehicle coordinate frame*. The intrinsic calibration of a sensor aims at recovering its specific internal parameters such as the focal length of a camera or the bias of an inertial measurement unit (IMU). We adopt a data-driven and self-supervised approach, *i.e.*, the sensors get continuously calibrated without human intervention while gathering data during normal operation. Along these lines, we automate the process of selecting measurements that are necessary for the successful calibration of the sensors. Furthermore, by separating the calibration parameters into observable and unobservable parts, we can safely use the incoming measurements to only perform updates in the observable subspace. This turns our algorithm into an online and long-term framework that could be smoothly deployed into autonomous systems.

Due to the wide field of applications of our approach, we will provide an abstract and high-level presentation. Specific applications will be demonstrated in Chap. 4. We assume the reader is acquainted with the mathematical formalism of Chap. 2, which shall be occasionally restated to ease understanding. After a brief introduction to the underlying probabilistic model, we will focus on the crucial parts of its inference and estimation with a particular attention to observability issues. As our algorithm relies on offline batch estimation techniques, we shall then suggest an information-theoretic scheme that allows online and long-term operations. The chapter terminates with practical aspects which should facilitate the implementation of our algorithm.

3.1 Problem Statement

The first step of our calibration procedure is to identify the different variables in the system and establish a probabilistic model through which they interact. These variables shall be classified in the following three categories:

1. Measurable random variable denoted by $\mathbf{X} = (X_1, \ldots, X_D)^T$ and samples thereof as $\mathbf{X}_i$. A realization of $\mathbf{X}_i$ corresponds to the measurement of a sensor $\mathcal{S}$.
2. Calibration variable denoted by $\boldsymbol{\Theta} = (\Theta_1, \ldots, \Theta_K)^T$. This latent random variable is the primary object of interest and comprises all the intrinsic and extrinsic parameters of the sensor $\mathcal{S}$.
3. Nuisance variable represented by $\boldsymbol{\Psi} = (\Psi_1, \ldots, \Psi_L)^T$. This latent random variable is not of direct interest, but the data distribution depends on it (*e.g.*, poses of landmarks).

For the sake of clarity, we have restricted here our formulation to a single sensor $\mathcal{S}$. As we shall demonstrate in Chap. 4, our approach smoothly generalizes to multiple sensors. Following the formalism of Sec. 2.1, we denote random variates of $\mathbf{X}$ by $\mathbf{x}$, of $\boldsymbol{\Theta}$ by $\boldsymbol{\theta}$, and of $\boldsymbol{\Psi}$ by $\boldsymbol{\psi}$. To remain generic, we assume that the random variables are multivariate.

All the variables being defined, we can introduce their joint probability density function $f_{\mathbf{X},\boldsymbol{\Theta},\boldsymbol{\Psi}}$ and the factorization

$$f_{\mathbf{X},\boldsymbol{\Theta},\boldsymbol{\Psi}}(\mathbf{x},\boldsymbol{\theta},\boldsymbol{\psi}) \;=\; f_{\mathbf{X}}(\mathbf{x} \mid \boldsymbol{\theta},\boldsymbol{\psi}) f_{\boldsymbol{\Theta},\boldsymbol{\Psi}}(\boldsymbol{\theta},\boldsymbol{\psi}). \tag{3.1}$$

Henceforth, we shall adopt the simplified notation of Equ. 3.1 for the conditional probability density functions, *i.e.*, omitting the random variables assignments on the conditional side. Assuming an *i.i.d.* data sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$ and one of its realization $\mathbf{x}_1, \ldots, \mathbf{x}_N$, the posterior joint probability density function over the latent variables is

$$f_{\boldsymbol{\Theta},\boldsymbol{\Psi}}(\boldsymbol{\theta},\boldsymbol{\psi} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N) \;=\; \frac{\prod_{i=1}^{N} f_{\mathbf{X}}(\mathbf{x}_i \mid \boldsymbol{\theta},\boldsymbol{\psi}) f_{\boldsymbol{\Theta},\boldsymbol{\Psi}}(\boldsymbol{\theta},\boldsymbol{\psi})}{\prod_{i=1}^{N} f_{\mathbf{X}}(\mathbf{x}_i)}. \tag{3.2}$$

As our final goal is to perform inference on the calibration variable $\boldsymbol{\Theta}$, we shall marginalize over the nuisance parameter $\boldsymbol{\Psi}$ to get the posterior marginal density function

$$f_{\boldsymbol{\Theta}}(\boldsymbol{\theta} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N) \;=\; \int_{\Omega_{\boldsymbol{\Psi}}} f_{\boldsymbol{\Theta},\boldsymbol{\Psi}}(\boldsymbol{\theta},\boldsymbol{\psi} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N)\, \mathrm{d}\boldsymbol{\psi}. \tag{3.3}$$

From the large sample theory, the posterior density in Equ. 3.2 can be approximated through Laplace's method by the normal distribution

$$f_{\boldsymbol{\Theta},\boldsymbol{\Psi}}(\boldsymbol{\theta},\boldsymbol{\psi} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N) \;\approx\; \mathcal{N}(\widehat{\boldsymbol{\theta}\boldsymbol{\psi}}, \mathcal{I}^{-1}(\widehat{\boldsymbol{\theta}\boldsymbol{\psi}})), \tag{3.4}$$

where

$$\widehat{\boldsymbol{\theta}\boldsymbol{\psi}} \;=\; \mathrm{Mode}\left[f_{\boldsymbol{\Theta},\boldsymbol{\Psi}}(\boldsymbol{\theta},\boldsymbol{\psi} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N)\right], \tag{3.5}$$

and $\mathcal{I}(\widehat{\boldsymbol{\theta}\boldsymbol{\psi}})$ is the observed Fisher information matrix at $\widehat{\boldsymbol{\theta}\boldsymbol{\psi}}$, *i.e.*, the negative of the Hessian matrix of the log-likelihood function evaluated at the joint posterior mode.

We denote the mean of the joint posterior density in Equ. 3.4 by $\boldsymbol{\mu}_{\boldsymbol{\Theta}\boldsymbol{\Psi}}$ and the covariance by $\boldsymbol{\Sigma}_{\boldsymbol{\Theta}\boldsymbol{\Psi}}$. Furthermore, we partition the mean as

$$\boldsymbol{\mu}_{\boldsymbol{\Theta}\boldsymbol{\Psi}} \quad = \quad \begin{pmatrix} \boldsymbol{\mu}_{\boldsymbol{\Theta}} \\ \boldsymbol{\mu}_{\boldsymbol{\Psi}} \end{pmatrix}, \tag{3.6}$$

and the covariance matrix as

$$\boldsymbol{\Sigma}_{\boldsymbol{\Theta}\boldsymbol{\Psi}} \quad = \quad \begin{pmatrix} \boldsymbol{\Sigma}^{*}_{\boldsymbol{\Theta}\boldsymbol{\Theta}} & \boldsymbol{\Sigma}^{*}_{\boldsymbol{\Theta}\boldsymbol{\Psi}} \\ \boldsymbol{\Sigma}^{*}_{\boldsymbol{\Psi}\boldsymbol{\Theta}} & \boldsymbol{\Sigma}^{*}_{\boldsymbol{\Psi}\boldsymbol{\Psi}} \end{pmatrix}. \tag{3.7}$$

Using the properties of the normal distribution presented in Sec. 2.1.3, we can derive a closed-form expression for the posterior marginal density in Equ. 3.3 as

$$f_{\boldsymbol{\Theta}}(\boldsymbol{\theta} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N) \quad \approx \quad \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\Theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\Theta}}), \tag{3.8}$$

where $\boldsymbol{\Sigma}_{\boldsymbol{\Theta}} = \boldsymbol{\Sigma}^{*}_{\boldsymbol{\Theta}\boldsymbol{\Theta}}$.

To conclude the mathematical formulation of our calibration method, we shall state the assumed form of the data distribution in Equ. 3.1 as

$$\mathbf{X} \quad \sim \quad \mathcal{N}(h(\boldsymbol{\theta}, \boldsymbol{\psi}), \boldsymbol{\Sigma}_{\mathbf{X}}), \tag{3.9}$$

where $h(\cdot)$ is an arbitrary twice-differentiable function and $\boldsymbol{\Sigma}_{\mathbf{X}}$ the known covariance matrix of the measurements. According to the discussions in Chap. 2, we have selected here a normal model. It can be justified by the central limit theorem, the principle of maximum entropy, and mostly by mathematical convenience. As we have demonstrated in Sec. 2.1.5, robust estimation techniques can also cope well with data violating the normal assumption.

3.2 Estimation and Inference

The calibration model outlined in Sec. 3.1 corresponds to the regression models of Sec. 2.1.6. We could therefore employ the same techniques to arrive at the joint posterior density in Equ. 3.2 and marginalize to get the marginal posterior density in Equ. 3.8. However, in robotic applications, the Jacobian matrix is likely to be a large *sparse matrix*. While $\boldsymbol{\Theta}$ is mostly low dimensional, $\boldsymbol{\Psi}$ can become high dimensional in a simultaneous localization and mapping (SLAM) scenario. Consequently, the Fisher information matrix and the covariance matrix are large *dense matrices*, which makes them intractable to compute or even store in memory. Furthermore, when using the rank-revealing matrix decompositions of Sec. 2.2.2, we want to be very accurate with the determination of the numerical rank. While the SVD would be the optimal tool, only an RRQR decomposition is applicable for these large matrices. Therefore, we propose in this section a method inspired by the Schur complement that first solves the least-squares problem for $\boldsymbol{\Theta}$ using an SVD and then calculate an estimate for $\boldsymbol{\Psi}$ with

an RRQR decomposition. Eventually, only the low-dimensional $\Sigma_{\boldsymbol{\Theta}}$ has to be computed from the SVD. It should be noted that we adopt an objective Bayesian viewpoint by setting an uninformative prior density $f_{\boldsymbol{\Theta},\boldsymbol{\Psi}}(\boldsymbol{\theta},\boldsymbol{\psi}) \propto 1$. The joint posterior mode in Equ. 3.5 hence coincides with the maximum-likelihood estimate provided by the least-squares method.

In the Jacobian matrix of Equ. 2.83, each column corresponds to one dimension in the parameter space. For our application, we shall thus partition it in the following manner

$$\mathbf{J} \quad = \quad \begin{pmatrix} \mathbf{J}_{\boldsymbol{\psi}} & \mathbf{J}_{\boldsymbol{\theta}} \end{pmatrix}, \tag{3.10}$$

where $\mathbf{J}_{\boldsymbol{\psi}} \in \mathbb{R}^{M \times L}$ corresponds to the Jacobian matrix with respect to $\boldsymbol{\psi}$, $\mathbf{J}_{\boldsymbol{\theta}} \in \mathbb{R}^{M \times K}$ with respect to $\boldsymbol{\theta}$, and $M = N \times D$ for a data sample of size N. For the sake of clarity, we hence assume that the data covariance matrix $\boldsymbol{\Sigma}$ has been absorbed in the Jacobian matrix and the right-hand side of the normal equations using the Cholesky decomposition in Equ. 2.88. Furthermore, we hide intentionally the column scaling of Equ. 2.133. Given these assumptions, the normal equations in Equ. 2.82 or Equ. 2.95 can be rewritten as

$$\begin{pmatrix} \mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\psi}} & \mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\theta}} \\ (\mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\theta}})^T & \mathbf{J}_{\boldsymbol{\theta}}^T \mathbf{J}_{\boldsymbol{\theta}} \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{\psi} \\ \Delta\boldsymbol{\theta} \end{pmatrix} \quad = \quad \begin{pmatrix} \mathbf{J}_{\boldsymbol{\psi}}^T \Delta\mathbf{T} \\ \mathbf{J}_{\boldsymbol{\theta}}^T \Delta\mathbf{T} \end{pmatrix}. \tag{3.11}$$

Without loss of generality, we have considered in Equ. 3.11 the nonlinear least-squares case and integrated the negation sign in the residual vector.

3.2.1 Calibration Variable Estimation

In order to isolate the subsystem of equations related to $\Delta\boldsymbol{\theta}$ only, we need to cancel out the lower-left block of the Fisher information matrix by left-multiplying it with the full-rank matrix

$$\mathbf{S} \quad = \quad \begin{pmatrix} \mathbf{I}_{\boldsymbol{\psi}} & \mathbf{0} \\ -(\mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\theta}})^T (\mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\psi}})^{-1} & \mathbf{I}_{\boldsymbol{\theta}} \end{pmatrix}, \tag{3.12}$$

where $\mathbf{I}_{\boldsymbol{\psi}} \in \mathbb{R}^{L \times L}$ and $\mathbf{I}_{\boldsymbol{\theta}} \in \mathbb{R}^{K \times K}$ are identity matrices. Left-multiplying the left-hand side of Equ. 3.11 by $\mathbf{S}$ yields

$$\begin{pmatrix} \mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\psi}} & \mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\theta}} \\ \mathbf{0} & \mathbf{J}_{\boldsymbol{\theta}}^T \mathbf{J}_{\boldsymbol{\theta}} - (\mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\theta}})^T (\mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\psi}})^{-1} \mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\theta}} \end{pmatrix}, \tag{3.13}$$

and the right-hand side gives

$$\begin{pmatrix} \mathbf{J}_{\boldsymbol{\psi}}^T \Delta\mathbf{T} \\ \mathbf{J}_{\boldsymbol{\theta}}^T \Delta\mathbf{T} - (\mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\theta}})^T (\mathbf{J}_{\boldsymbol{\psi}}^T \mathbf{J}_{\boldsymbol{\psi}})^{-1} \mathbf{J}_{\boldsymbol{\psi}}^T \Delta\mathbf{T} \end{pmatrix}. \tag{3.14}$$

It follows from Equ. 3.13 that we can now solve the equations independently for $\Delta\boldsymbol{\theta}$.

We shall further investigate how to efficiently compute the matrix expressions in the lower parts of Equ. 3.13 and Equ. 3.14. In particular, we shall focus on $(\mathbf{J}_{\boldsymbol{\psi}}^T\mathbf{J}_{\boldsymbol{\theta}})^T(\mathbf{J}_{\boldsymbol{\psi}}^T\mathbf{J}_{\boldsymbol{\psi}})^{-1}\mathbf{J}_{\boldsymbol{\psi}}^T$, which can rapidly become intractable. To this end, we first apply the RRQR decomposition in Equ. 2.136 to $\mathbf{J}_{\boldsymbol{\psi}}$, which leads to

$$\mathbf{J}_{\boldsymbol{\psi}} = \mathbf{Q}\mathbf{R}\boldsymbol{\Pi}^T, \tag{3.15}$$

and we use it to simplify the matrix expression as follows

$$\begin{aligned}
&(\mathbf{J}_{\boldsymbol{\psi}}^T\mathbf{J}_{\boldsymbol{\theta}})^T(\mathbf{J}_{\boldsymbol{\psi}}^T\mathbf{J}_{\boldsymbol{\psi}})^{-1}\mathbf{J}_{\boldsymbol{\psi}}^T \\
&= \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}\mathbf{R}\boldsymbol{\Pi}^T(\boldsymbol{\Pi}\mathbf{R}^T\mathbf{Q}^T\mathbf{Q}\mathbf{R}\boldsymbol{\Pi}^T)^{-1}\boldsymbol{\Pi}\mathbf{R}^T\mathbf{Q}^T \\
&= \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}\mathbf{R}\boldsymbol{\Pi}^T(\boldsymbol{\Pi}\mathbf{R}^T\mathbf{R}\boldsymbol{\Pi}^T)^{-1}\boldsymbol{\Pi}\mathbf{R}^T\mathbf{Q}^T \\
&= \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1\mathbf{R}_1\boldsymbol{\Pi}^T(\boldsymbol{\Pi}\mathbf{R}_1^T\mathbf{R}_1\boldsymbol{\Pi}^T)^{-1}\boldsymbol{\Pi}\mathbf{R}_1^T\mathbf{Q}_1^T \\
&= \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1\mathbf{R}_1\boldsymbol{\Pi}^T\boldsymbol{\Pi}\mathbf{R}_1^{-1}\mathbf{R}_1^{-T}\boldsymbol{\Pi}^T\boldsymbol{\Pi}\mathbf{R}_1^T\mathbf{Q}_1^T \\
&= \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1\mathbf{R}_1\mathbf{R}_1^{-1}\mathbf{R}_1^{-T}\mathbf{R}_1^T\mathbf{Q}_1^T \\
&= \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1\mathbf{Q}_1^T,
\end{aligned} \tag{3.16}$$

where $\mathbf{R}_1 \in \mathbb{R}^{L\times L}$ is the upper-triangular part of $\mathbf{R} \in \mathbb{R}^{M\times L}$ and $\mathbf{Q}_1 \in \mathbb{R}^{M\times L}$ contains the first L Householder vectors of $\mathbf{Q} \in \mathbb{R}^{M\times M}$. In this derivation, we have first used the orthogonality of $\mathbf{Q}$ to eliminate it from the matrix inversion. In a second step, we have converted the decomposition into a thin QR decomposition. Lastly, the orthogonality of $\boldsymbol{\Pi}$ has allowed us to arrive at the final result. Consequently, we have transformed an involved expression containing a large matrix inversion into a much affordable matrix multiplication. Furthermore, we are not affected by the numerical rank of $\mathbf{J}_{\boldsymbol{\psi}}$ since we do not compute any matrix inverse, and we only need a reduced QR decomposition.

Using Equ. 3.16, we introduce the following auxiliary variables

$$\mathbf{A}_{\boldsymbol{\theta}} = \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{J}_{\boldsymbol{\theta}} - (\mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1)(\mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1)^T, \tag{3.17}$$

$$\mathbf{b}_{\boldsymbol{\theta}} = \mathbf{J}_{\boldsymbol{\theta}}^T\Delta\mathbf{T} - (\mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1)(\Delta\mathbf{T}^T\mathbf{Q}_1)^T, \tag{3.18}$$

such that the subsystem of equations for $\Delta\boldsymbol{\theta}$ becomes

$$\mathbf{A}_{\boldsymbol{\theta}}\Delta\boldsymbol{\theta} = \mathbf{b}_{\boldsymbol{\theta}}, \tag{3.19}$$

where $\mathbf{A}_{\boldsymbol{\theta}} \in \mathbb{R}^{K\times K}$ and $\mathbf{b}_{\boldsymbol{\theta}} \in \mathbb{R}^K$. When using the Householder form of $\mathbf{Q}_1$, $\mathbf{A}_{\boldsymbol{\theta}}$ and $\mathbf{b}_{\boldsymbol{\theta}}$ can be computed at low cost. Moreover, even though $\mathbf{A}_{\boldsymbol{\theta}}$ is dense, the small size of K keeps the memory footprint low. Prior to solving these equations, we factorize $\mathbf{A}_{\boldsymbol{\theta}}$ using the SVD from Equ. 2.113 such that

$$\mathbf{A}_{\boldsymbol{\theta}} = \mathbf{U}\mathbf{S}\mathbf{V}^T. \tag{3.20}$$

Following the methodology from Sec. 2.2.2, we compute the numerical rank $r_{\epsilon_{\boldsymbol{\theta}}}$ of $\mathbf{A}_{\boldsymbol{\theta}}$ from its singular values using a threshold $\epsilon_{\boldsymbol{\theta}}$. This threshold is proportional to the magnitude of the noise in $\mathbf{A}_{\boldsymbol{\theta}}$ and we shall further investigate its

determination in Sec. 3.4. From the SVD and the numerical rank, we can readily solve Equ. 3.19 using a TSVD as

$$\Delta\boldsymbol{\theta} = \sum_{i=1}^{r_{\epsilon_{\boldsymbol{\theta}}}} \frac{\mathbf{u}_i^T \mathbf{b}_{\boldsymbol{\theta}}}{\sigma_i} \mathbf{v}_i. \tag{3.21}$$

The computation of the full covariance matrix involves the inverse of the Fisher information matrix. However, for a 2×2 block matrix as in the left-hand side of Equ. 3.11, we can advantageously perform blockwise inversions. Due to the marginalization property of the normal distribution, we only require the lower-right block given by $\mathbf{A}_{\boldsymbol{\theta}}^{\dagger}$, which yields the marginal covariance matrix

$$\boldsymbol{\Sigma}_{\boldsymbol{\Theta}} = \sum_{i=1}^{r_{\epsilon_{\boldsymbol{\theta}}}} \frac{\mathbf{v}_i \mathbf{u}_i^T}{\sigma_i}. \tag{3.22}$$

Furthermore, we have a direct access to the numerical nullspace and column space of $\mathbf{A}_{\boldsymbol{\theta}}$ as

$$\text{null}\left[\mathbf{A}_{\boldsymbol{\theta}}, \epsilon_{\boldsymbol{\theta}}\right] = \text{span}\left[\left\{\mathbf{v}_{r_{\epsilon_{\boldsymbol{\theta}}}+1}, \ldots, \mathbf{v}_K\right\}\right], \tag{3.23}$$

$$\text{range}\left[\mathbf{A}_{\boldsymbol{\theta}}, \epsilon_{\boldsymbol{\theta}}\right] = \text{span}\left[\left\{\mathbf{u}_1, \ldots, \mathbf{u}_{r_{\epsilon_{\boldsymbol{\theta}}}}\right\}\right]. \tag{3.24}$$

The basis vectors from the numerical nullspace correspond to directions in the parameter space which are numerically unobservable. While the first $r_{\epsilon_{\boldsymbol{\theta}}}$ column vectors of $\mathbf{V}$ form an orthonormal basis for the row space of $\mathbf{A}_{\boldsymbol{\theta}}$ or the *observable subspace*, the basis vectors from the nullspace form an orthonormal basis for the *unobservable subspace*.

If we stack the basis vectors of the nullspace into a matrix $\mathbf{N}_{\mathbf{A}_{\boldsymbol{\theta}}} \in \mathbb{R}^{K \times (K - r_{\epsilon_{\boldsymbol{\theta}}})}$, we reveal the local observability structure of the parameter. Each row i of the nullspace matrix represents the observability of the dimension i of the parameter vector. A row of zeros implies that the corresponding dimension is fully observable. A row containing a single one signifies that the corresponding dimension is unobservable. Any other rows correspond to dimensions in the parameter space that are only observable as a linear combination of other dimensions. We can interpret the TSVD solution in Equ. 3.21 as the unique least-squares solution in the observable subspace. In other words, the shift vector will only update the observable part of the parameter vector.

Instead of inspecting the nullspace of $\mathbf{A}_{\boldsymbol{\theta}}$, we can construct the row space matrix $\mathbf{R}_{\mathbf{A}_{\boldsymbol{\theta}}} \in \mathbb{R}^{K \times r_{\epsilon_{\boldsymbol{\theta}}}}$. As earlier, a row i of $\mathbf{R}_{\mathbf{A}_{\boldsymbol{\theta}}}$ corresponds to a dimension i of $\boldsymbol{\theta}$. Hence, the 2-norm of row i defines a measure $O(\theta_i) \in [0, 1]$, the columns of $\mathbf{V}^T$ forming an orthonormal basis, that quantifies the observability of θ_i given the current data.

3.2.2 Nuisance Variable Estimation

For a linear system, Equ. 3.21 and Equ. 3.22 directly provide the mean and the covariance of the posterior marginal density in Equ. 3.8. In the nonlinear least-squares case, once a solution is found for $\Delta\boldsymbol{\theta}$, we also need to compute

$\Delta\boldsymbol{\psi}$ for the next iteration of the optimization method. At the convergence of the algorithm, the mean is given by the resulting local minimum for $\boldsymbol{\theta}$ and the covariance by Equ. 3.22.

Plugging $\Delta\boldsymbol{\theta}$ into Equ. 3.11, $\Delta\boldsymbol{\psi}$ becomes the solution to the following system of equations

$$(\mathbf{J}_{\boldsymbol{\psi}}^T\mathbf{J}_{\boldsymbol{\psi}})\Delta\boldsymbol{\psi} \quad = \quad \mathbf{J}_{\boldsymbol{\psi}}^T(\Delta\mathbf{T} - \mathbf{J}_{\boldsymbol{\theta}}\Delta\boldsymbol{\theta}), \tag{3.25}$$

which can be recognized as the normal equations of the least-squares problem

$$\min_{\Delta\boldsymbol{\psi}\in\mathbb{R}^L} \left\|\mathbf{J}_{\boldsymbol{\psi}}\Delta\boldsymbol{\psi} - (\Delta\mathbf{T} - \mathbf{J}_{\boldsymbol{\theta}}\Delta\boldsymbol{\theta})\right\|_2 . \tag{3.26}$$

Given the potential large size of $\mathbf{J}_{\boldsymbol{\psi}}$, we solve Equ. 3.26 with an RRQR decomposition. In most applications, this is indeed the only possibility. Furthermore, this decomposition is already available from Equ. 3.15 and we directly compute a basic solution as in Equ. 2.159, *i.e.*,

$$\Delta\boldsymbol{\psi} \quad = \quad \boldsymbol{\Pi}\begin{pmatrix} \mathbf{R}_{11}^{-1}\mathbf{Q}_{11}^T(\Delta\mathbf{T} - \mathbf{J}_{\boldsymbol{\theta}}\Delta\boldsymbol{\theta}) \\ \mathbf{0} \end{pmatrix}, \tag{3.27}$$

where $\mathbf{Q}_{11} \in \mathbb{R}^{M\times r_{\epsilon_{\boldsymbol{\psi}}}}$ contains the first $r_{\epsilon_{\boldsymbol{\psi}}}$ Householder vectors of $\mathbf{Q}$, $\mathbf{R}_{11} \in \mathbb{R}^{r_{\epsilon_{\boldsymbol{\psi}}}\times r_{\epsilon_{\boldsymbol{\psi}}}}$ is the upper-left triangular part of $\mathbf{R}$, and $r_{\epsilon_{\boldsymbol{\psi}}}$ the estimated numerical rank of $\mathbf{J}_{\boldsymbol{\psi}}$ with tolerance $\epsilon_{\boldsymbol{\psi}}$. Instead of inverting $\mathbf{R}_{11}$, we solve the system by back substitution. The basic solution has at most $r_{\epsilon_{\boldsymbol{\psi}}}$ nonzero components. If $\mathbf{J}_{\boldsymbol{\psi}}$ has full rank, the lower null vector disappears and Equ. 3.27 yields the unique least-squares solution to Equ. 3.26. In this thesis, we have chosen the basic solution for its computational simplicity. The least-squares solution would have required the complete orthogonal decomposition outlined in Equ. 2.157. As our primary interest lies in estimating $\boldsymbol{\theta}$, we believe that the basic solution is suitable for $\boldsymbol{\psi}$. This claim shall further be supported by the experimental evaluation in Chap. 4.

3.2.3 Improper Posterior Inference

If the calibration parameter is partially observable or unobservable, the posterior marginal density in Equ. 3.8 is improper. The covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\Theta}}$ is singular and the posterior is a *degenerate normal distribution*. However, we can still define proper density functions in the observable and unobservable subspaces. To this end, we shall first apply the following affine transformation to $\boldsymbol{\Theta}$

$$\widetilde{\boldsymbol{\Theta}} \quad = \quad \mathbf{V}^T\boldsymbol{\Theta}, \tag{3.28}$$

where $\mathbf{V}$ is the orthogonal matrix from the SVD in Equ. 3.20. Due to the affine transformation property of the normal distribution, the random vector $\widetilde{\boldsymbol{\Theta}}$ is normally distributed as

$$\widetilde{\boldsymbol{\Theta}} \mid \mathbf{x}_1, \ldots, \mathbf{x}_n \quad \sim \quad \mathcal{N}(\mathbf{V}^T\boldsymbol{\mu}_{\boldsymbol{\Theta}}, \mathbf{V}^T\boldsymbol{\Sigma}_{\boldsymbol{\Theta}}\mathbf{V}). \tag{3.29}$$

We can further partition $\widetilde{\boldsymbol{\Theta}}$ into an observable and unobservable part $\widetilde{\boldsymbol{\Theta}} = (\widetilde{\boldsymbol{\Theta}}_{obs}, \widetilde{\boldsymbol{\Theta}}_{nobs})^T$, where

$$\begin{aligned} \widetilde{\boldsymbol{\Theta}}_{obs} &= (\widetilde{\Theta}_1, \ldots, \widetilde{\Theta}_{r_{\epsilon_{\boldsymbol{\theta}}}})^T, \\ \widetilde{\boldsymbol{\Theta}}_{nobs} &= (\widetilde{\Theta}_{r_{\epsilon_{\boldsymbol{\theta}}}+1}, \ldots, \widetilde{\Theta}_K)^T. \end{aligned} \tag{3.30}$$

We adopt the same partition for the mean vector of $\widetilde{\boldsymbol{\Theta}}$, *i.e.*,

$$\boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}} = \mathbf{V}^T \boldsymbol{\mu}_{\boldsymbol{\Theta}} = (\boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{obs}}, \boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{nobs}})^T. \tag{3.31}$$

Finally, we can use the orthogonality of $\mathbf{V}$ to simplify the covariance matrix of $\widetilde{\boldsymbol{\Theta}}$ into

$$\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\Theta}}} = \mathbf{V}^T \boldsymbol{\Sigma}_{\boldsymbol{\Theta}} \mathbf{V} = \mathbf{S}^{\dagger}, \tag{3.32}$$

where $\mathbf{S}$ is the diagonal matrix of the singular values of $\mathbf{A}_{\boldsymbol{\theta}}$ defined in Equ. 3.20 and its pseudoinverse is the $K \times K$ matrix

$$\mathbf{S}^{\dagger} = \operatorname{diag}\left[\sigma_1^{-1}, \ldots, \sigma_{r_{\epsilon_{\boldsymbol{\theta}}}}^{-1}\right]. \tag{3.33}$$

Using the marginalization property of the normal distribution, the observable part is normally distributed as

$$\widetilde{\boldsymbol{\Theta}}_{obs} \mid \mathbf{x}_1, \ldots, \mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{obs}}, \mathbf{S}^{\dagger}_{r_{\epsilon_{\boldsymbol{\theta}}}}), \tag{3.34}$$

where $\mathbf{S}^{\dagger}_{r_{\epsilon_{\boldsymbol{\theta}}}} \in \mathbb{R}^{r_{\epsilon_{\boldsymbol{\theta}}} \times r_{\epsilon_{\boldsymbol{\theta}}}}$ is the upper-left nonzero block of $\mathbf{S}^{\dagger}$. Lastly, the unobservable part is distributed as

$$\widetilde{\boldsymbol{\Theta}}_{nobs} \mid \mathbf{x}_1, \ldots, \mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{nobs}}, \mathbf{0}), \tag{3.35}$$

which can be interpreted as a *Dirac distribution* [Mur12] with density given by

$$\begin{aligned} f_{\widetilde{\boldsymbol{\Theta}}_{nobs}}(\widetilde{\boldsymbol{\theta}}_{nobs} \mid \boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{nobs}}) &= \delta(\widetilde{\boldsymbol{\theta}}_{nobs} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{nobs}}), \\ \delta(\widetilde{\boldsymbol{\theta}}_{nobs} - \boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{nobs}}) &= \begin{cases} +\infty & \text{for } \widetilde{\boldsymbol{\theta}}_{nobs} = \boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{nobs}} \\ 0 & \text{for } \widetilde{\boldsymbol{\theta}}_{nobs} \neq \boldsymbol{\mu}_{\widetilde{\boldsymbol{\Theta}}_{nobs}}. \end{cases} \end{aligned} \tag{3.36}$$

3.3 Online Inference

Statistical researchers and data analysts are rarely affected by lengthy datasets. In these fields, a potentially huge amount of data is collected for offline analysis that can last days or weeks on powerful computers. Indeed, few of the referenced

statistical books in our bibliography discuss the matter of online estimation. On the other hand, online learning is a key element for robotic applications, *e.g.*, for state estimation. It has always been a dream to deploy machines in the nature that can continuously learn from the environment without human intervention in a long-term basis. To this end, Bayes's theorem is amenable to recursive estimation. For each new data point, the posterior of the previous step becomes the prior for the current step. A dataset can thus be processed sequentially and the posterior density is available at each timestep. In a robot localization scenario, this sequential application of Bayes's theorem has been coined *Bayes filter*. As briefly mentioned in Sec. 2.1.3, Bayesian inference becomes intractable for arbitrary density functions and sampling methods are required. For the particular case of a linear-normal model, a closed-form solution can be efficiently calculated, *e.g.*, with the *Kalman filter (KF)*. In the nonlinear-normal case, various extensions to the KF have been proposed, for instance, the *extended Kalman filter (EKF)* which uses a linear approximation of the densities around the current estimate, similar to what we have presented in Sec. 2.1.6. However, the EKF does not iterate to refine the linearization and a batch method might in that regard be more accurate. Furthermore, the KF and EKF require an informative prior to start with.

In this thesis, we want to benefit from the robustness of offline methods, along with the possibility to use our algorithm in an online and long-term fashion. In a more philosophical sense, human intelligence can be a great source of inspiration. Humans do not simply discard information sequentially. We store memories and mix them up with fresh knowledge to take decisions. As new information arrives, we continuously decide whether to keep it based on its utility, eventually replacing or refining old knowledge. Our calibration algorithm attempts to partially imitate this behavior. We process batches of data sequentially and decide to keep them based on their utility for the calibration. Each new batch is merged to the old ones to refine our knowledge about the calibration parameters until we reach a satisfactory level of confidence.

To start with the mathematical formulation of our online method, we define the current set of stored data samples as $\mathcal{D}^{info} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, which has led to the posterior marginal density in Equ. 3.8, associated with the random variable $\boldsymbol{\Theta} \mid \mathcal{D}^{info}$. The set $\mathcal{D}^{info}$ contains the current *informative measurements* for the calibration variable. Our sensor $\mathcal{S}$ continuously streams new data that we accumulate in another batch of size ΔN denoted by $\mathcal{D}^{new} = \{\mathbf{x}_{N+1}, \dots, \mathbf{x}_{N+\Delta N}\}$. Intuitively, if the measurements in $\mathcal{D}^{new}$ are identical to those in $\mathcal{D}^{info}$, we are not really improving our knowledge about $\boldsymbol{\Theta}$ and we could as well discard $\mathcal{D}^{new}$.

The information theory presented in Sec. 2.1.2 provides a principled way to evaluate the usefulness of $\mathcal{D}^{new}$. Shortly after its introduction in the signal processing community, information theory has been adopted by statisticians, notably by Bayesian proponents such as Lindley. In [Lin56], he defines the amount of information in a random variable as the negative Shannon entropy. A random variable with high entropy, *e.g.* uniformly distributed, contains little information. Conversely, a sharply-peaked distribution with low entropy conveys high information. The amount of information provided by an experiment may therefore be defined as the difference between the posterior and prior information. This can be interpreted as a measure of the *utility* or *information gain* of the experiment. In our setup, we can consider the prior to be $\boldsymbol{\Theta} \mid \mathcal{D}^{info}$, the posterior $\boldsymbol{\Theta} \mid \mathcal{D}^{info}, \mathcal{D}^{new}$, and define the utility to be

$$\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new}) \quad = \quad h(\boldsymbol{\Theta} \mid \mathcal{D}^{info}) - h(\boldsymbol{\Theta} \mid \mathcal{D}^{info}, \mathcal{D}^{new}), \quad (3.37)$$

where $h(\cdot)$ denotes the differential entropy (*cf.* Equ. 2.22). It follows from Equ. 3.37 that if $\mathcal{D}^{new}$ is useful for estimating $\boldsymbol{\Theta}$, it reduces its entropy and increases its information. In other words, the utility tells us whether it is worth adding $\mathcal{D}^{new}$ to our set of informative measurements. If the utility falls under a prescribed threshold δ, $\mathcal{D}^{new}$ can be discarded since it does not bring a significant information gain. Otherwise, it can be added to the informative set which becomes $\mathcal{D}^{info} \cup \mathcal{D}^{new}$. As mentioned in Sec. 2.1.2, the differential entropy is not a proper measure of information. Therefore, the Kullback-Leibler (KL) divergence defined in Equ. 2.28 provides an alternative definition of the information gain as

$$\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new}) \quad = \quad \mathcal{D}_{KL}(\boldsymbol{\Theta} \mid \mathcal{D}^{info}, \mathcal{D}^{new} \;||\; \boldsymbol{\Theta} \mid \mathcal{D}^{info}). \quad (3.38)$$

The KL divergence remains well-defined for continuous random variables and can also be interpreted as the information loss when approximating the posterior with the prior. It is worth noting that the utility is a function of the observed data $\mathcal{D}^{new}$. If we take the expectation of the utility over the data sample space, we retrieve the mutual information defined in Equ. 2.27, *i.e.*,

$$\mathbb{E}_{\mathcal{D}^{new}}\left[\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new})\right] \quad = \quad I(\boldsymbol{\Theta} \mid \mathcal{D}^{info}; \mathcal{D}^{new}). \quad (3.39)$$

As pointed out by Lindley, the mutual information is also a well-behaved measure of information for continuous and discrete random variables. However, it remains intractable to compute in our setting and counterintuitive. As it is averaged over the data sample space, the mutual information can technically be calculated prior to actually observing the data. In our calibration framework, a decision has to be taken given the current observed data. Therefore, we adopt the utility function as a measure of information. As will be demonstrated in Chap. 4, this scheme is particularly useful for our application. As we accumulate data over time, we can optimally select the measurements that maximize the information in $\boldsymbol{\Theta}$. For cases of partial observability, the information gain converges to zero and increases when the system becomes fully observable.

We shall now examine the derivation of the utility function in our context where the random variables are normally distributed. The differential entropy of a normally distributed random variable $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{X}}, \boldsymbol{\Sigma}_{\mathbf{X}})$ of dimension K is given by

$$h(\mathbf{X}) \quad = \quad \frac{K}{2}(1 + \log_e(2\pi)) + \frac{1}{2}\log_e|\boldsymbol{\Sigma}_{\mathbf{X}}|, \quad (3.40)$$

where $|\cdot|$ denotes the *matrix determinant* and $\log_e$ the *natural logarithm*. This latter is chosen for mathematical convenience, given the exponential form of the normal distribution. We can retrieve an entropy in bits by dividing Equ. 3.40 by $\log_e(2)$. If we now state the distribution of the prior as $\boldsymbol{\Theta} \mid \mathcal{D}^{info} \sim \mathcal{N}(\boldsymbol{\mu}^{prior}, \boldsymbol{\Sigma}^{prior})$ and of the posterior as $\boldsymbol{\Theta} \mid \mathcal{D}^{info}, \mathcal{D}^{new} \sim \mathcal{N}(\boldsymbol{\mu}^{post}, \boldsymbol{\Sigma}^{post})$, the information gain in Equ. 3.37 becomes

$$\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new}) = \frac{1}{2} \log_e \frac{|\boldsymbol{\Sigma}^{prior}|}{|\boldsymbol{\Sigma}^{post}|}. \tag{3.41}$$

The SVD of the inverse of the covariance matrices $\boldsymbol{\Sigma}^{prior}$ and $\boldsymbol{\Sigma}^{post}$ are readily available from Equ. 3.20. If we denote these inverse matrices and their SVD by

$$\mathbf{A}_{\boldsymbol{\theta}}^{prior} = \mathbf{U}_b \mathbf{S}_b \mathbf{V}_b^T, \tag{3.42}$$
$$\mathbf{A}_{\boldsymbol{\theta}}^{post} = \mathbf{U}_a \mathbf{S}_a \mathbf{V}_a^T, \tag{3.43}$$

we can use the orthogonality of the singular vectors and simplify Equ. 3.41 to

$$\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new}) = \frac{1}{2} \log_e \frac{|\mathbf{S}_a|}{|\mathbf{S}_b|}. \tag{3.44}$$

Furthermore, as $\mathbf{S}_a = \text{diag}\left[\sigma_1^a, \ldots, \sigma_K^a\right]$ and $\mathbf{S}_b = \text{diag}\left[\sigma_1^b, \ldots, \sigma_K^b\right]$ are diagonal matrices, their determinant is the product of their singular values. Finally, when using the property of the logarithm, Equ. 3.44 becomes

$$\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new}) = \frac{1}{2} \left(\sum_{i=1}^{r_{\epsilon_{\boldsymbol{\theta}}}^a} \log_e \sigma_i^a - \sum_{i=1}^{r_{\epsilon_{\boldsymbol{\theta}}}^b} \log_e \sigma_i^b \right), \tag{3.45}$$

where $r_{\epsilon_{\boldsymbol{\theta}}}^a$ and $r_{\epsilon_{\boldsymbol{\theta}}}^b$ are the numerical ranks of $\mathbf{A}_{\boldsymbol{\theta}}^{post}$ and $\mathbf{A}_{\boldsymbol{\theta}}^{prior}$ respectively. In summary, the information gain in Equ. 3.45 is available at low cost from the already computed SVD. We can simply maintain the sum of the logarithm of the singular values of the current estimate.

We shall conclude this section with the derivation of the utility function in term of the KL divergence between the prior and the posterior. The distributions being normally distributed, there exists a closed-form solution given by

$$\begin{aligned} &\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new}) \\ &= \frac{1}{2} \log_e \frac{|\boldsymbol{\Sigma}^{prior}|}{|\boldsymbol{\Sigma}^{post}|} - \frac{K}{2} \\ &+ \frac{1}{2} \text{tr} \left((\boldsymbol{\Sigma}^{prior})^{-1} \boldsymbol{\Sigma}^{post} \right) \\ &+ \frac{1}{2} (\boldsymbol{\mu}^{prior} - \boldsymbol{\mu}^{post})^T (\boldsymbol{\Sigma}^{prior})^{-1} (\boldsymbol{\mu}^{prior} - \boldsymbol{\mu}^{post}), \end{aligned} \tag{3.46}$$

where the *trace* operator $\text{tr}(\cdot)$ returns the sum of the diagonal elements of its matrix argument. We witness that the KL divergence contains Equ. 3.41 and supplementary terms. As before, the inverse covariance matrix is already available from Equ. 3.20. The other operations are straightforward.

3.4 Implementation Details

After a rather formal presentation of our method, we want to conclude this chapter with some practical considerations. This section can be considered as a summary of our approach towards a functional realization. After a discussion on the sensor model, we will outline an algorithmic view of our framework, followed by a complexity analysis and some implementation details.

3.4.1 Sensor Model

For a given sensor $\mathcal{S}$, our method requires a probabilistic model describing the data generation process. In statistical terms, we need to specify the data distribution, which, as presented earlier, is in the family of normal distributions. Therefore, we have to state the *forward model* $h_{\mathcal{S}}(\boldsymbol{\theta}, \boldsymbol{\psi})$ such that for a data sample $\mathbf{X}_i$ of $\mathbf{X}$, the prediction error is the random vector

$$\mathbf{e}_i \quad = \quad \mathbf{X}_i - h_{\mathcal{S}}(\boldsymbol{\theta}, \boldsymbol{\psi}), \tag{3.47}$$

distributed as

$$\mathbf{e}_i \quad \sim \quad \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{X}}), \tag{3.48}$$

where $\boldsymbol{\Sigma}_{\mathbf{X}}$ is a deterministic covariance matrix. This latter is either available from the datasheet of the sensor $\mathcal{S}$ or should be estimated. An estimator $\widehat{\boldsymbol{\Sigma}}_{\mathbf{X}}$ can be derived from the sample covariance matrix (*cf.* Equ. 2.35) of the error terms $\mathbf{e}_i$ using initial guesses for the calibration variable $\boldsymbol{\theta}^{(0)}$ and the nuisance variable $\boldsymbol{\psi}^{(0)}$.

From a data sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$, the covariance matrices $\boldsymbol{\Sigma}_{\mathbf{X}_i}$ can be assembled into the $M \times M$ *block diagonal matrix*

$$\boldsymbol{\Sigma} \quad = \quad \operatorname{diag}\left[\boldsymbol{\Sigma}_{\mathbf{X}_1}, \ldots, \boldsymbol{\Sigma}_{\mathbf{X}_N}\right], \tag{3.49}$$

where $M = N \times D$. If we recall Equ. 2.89, we can use a simplified form of the normal equations by left-multiplying the Jacobian matrix and the residual vector by the Cholesky factor of $\boldsymbol{\Sigma}^{-1}$, which we denote here $\boldsymbol{\Sigma}^{-\frac{1}{2}}$. Taking into account the block diagonal structure of $\boldsymbol{\Sigma}$, this Cholesky factor can be expressed as

$$\boldsymbol{\Sigma}^{-\frac{1}{2}} \quad = \quad \operatorname{diag}\left[\boldsymbol{\Sigma}_{\mathbf{X}_1}^{-\frac{1}{2}}, \ldots, \boldsymbol{\Sigma}_{\mathbf{X}_N}^{-\frac{1}{2}}\right], \tag{3.50}$$

where the $\boldsymbol{\Sigma}_{\mathbf{X}_i}^{-\frac{1}{2}}$ are the Cholesky factors of the $\boldsymbol{\Sigma}_{\mathbf{X}_i}$.

If the data distribution violates the normal assumption, some of the error terms $\mathbf{e}_i$ shall be considered as outliers and consequently penalized. The robust estimation techniques presented in Sec. 2.1.5 provide a sound workaround. In this thesis, we adopt the strategy of the M-estimators, where for each $\mathbf{e}_i$, a weight $w(\mathbf{e}_i)$ smoothly dampens the effect of outliers. The weights are computed at each iteration of the nonlinear optimization and we use a modified Blake-Zisserman cost function, yielding for each error term

$$w(\mathbf{e}_i, \boldsymbol{\Sigma}_{\mathbf{X}_i}) = \frac{\exp(-\mathbf{e}_i^T \boldsymbol{\Sigma}_{\mathbf{X}_i}^{-1} \mathbf{e}_i)}{\exp(-\mathbf{e}_i^T \boldsymbol{\Sigma}_{\mathbf{X}_i}^{-1} \mathbf{e}_i) + \epsilon}, \tag{3.51}$$

where ϵ is a priori calculated using Equ. 2.75 with algorithmic parameters p and w_q. p (close to 1) is a probability that controls the boundary for outliers and w_q (close to 0) the damping factor. We henceforth group these parameters into the set $\mathcal{P}_M = \{p, w_q\}$. Once the weights are computed, we can multiply them into the matrix of Cholesky factors in Equ. 3.50 to get

$$\widetilde{\boldsymbol{\Sigma}}^{-\frac{1}{2}} = \operatorname{diag}\left[\widetilde{\boldsymbol{\Sigma}}_{\mathbf{X}_1}^{-\frac{1}{2}}, \ldots, \widetilde{\boldsymbol{\Sigma}}_{\mathbf{X}_N}^{-\frac{1}{2}}\right], \tag{3.52}$$

where

$$\widetilde{\boldsymbol{\Sigma}}_{\mathbf{X}_i}^{-\frac{1}{2}} = w^{\frac{1}{2}}(\mathbf{e}_i, \boldsymbol{\Sigma}_{\mathbf{X}_i}) \boldsymbol{\Sigma}_{\mathbf{X}_i}^{-\frac{1}{2}}. \tag{3.53}$$

The function $h_{\mathcal{S}}(\cdot)$ can be regarded as a predictor for the measurements of the sensor $\mathcal{S}$ and its design is part of an priori analysis. If variations of the input parameters to $h_{\mathcal{S}}(\cdot)$ do not change the output value, the system is not observable. Consequently, the data distribution is not identifiable. As debated earlier, the parameter vectors might nevertheless be partially observable, *i.e.*, only changes in a subspace of the parameters generate the same output. Accordingly, we shall be able to estimate the observable subspace of the parameter vectors from a sample of the data distribution.

We henceforth assume that the function $h_{\mathcal{S}}(\cdot)$ is nonlinear in the parameters. In case it is linear, we can easily replace the nonlinear optimization part of our algorithm by a linear least-squares method. The forward model should be differentiable in order to build the Jacobian matrices of the error terms $\mathbf{e}_i$ with respect to $\boldsymbol{\theta}$

$$\mathbf{J}_i(\boldsymbol{\theta}) = \frac{\partial \mathbf{e}_i}{\partial \boldsymbol{\theta}} \tag{3.54}$$

and to $\boldsymbol{\psi}$

$$\mathbf{J}_i(\boldsymbol{\psi}) = \frac{\partial \mathbf{e}_i}{\partial \boldsymbol{\psi}}. \tag{3.55}$$

At each step of the iterative optimization method, we require the evaluation of these Jacobian matrices at the current estimate for the parameters. The Jacobian matrices can be provided in an analytical form or calculated numerically, *e.g.*, using finite difference approximations. For the data sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$, the Jacobian matrices are stacked together and left-multiplied by $\widetilde{\boldsymbol{\Sigma}}^{-\frac{1}{2}}$, *i.e.*,

$$\mathbf{J}_{\boldsymbol{\theta}} = \begin{pmatrix} \widetilde{\boldsymbol{\Sigma}}_{\mathbf{X}_1}^{-\frac{1}{2}} \mathbf{J}_1(\boldsymbol{\theta}) \\ \vdots \\ \widetilde{\boldsymbol{\Sigma}}_{\mathbf{X}_N}^{-\frac{1}{2}} \mathbf{J}_N(\boldsymbol{\theta}) \end{pmatrix}, \quad \mathbf{J}_{\boldsymbol{\psi}} = \begin{pmatrix} \widetilde{\boldsymbol{\Sigma}}_{\mathbf{X}_1}^{-\frac{1}{2}} \mathbf{J}_1(\boldsymbol{\psi}) \\ \vdots \\ \widetilde{\boldsymbol{\Sigma}}_{\mathbf{X}_N}^{-\frac{1}{2}} \mathbf{J}_N(\boldsymbol{\psi}) \end{pmatrix}. \tag{3.56}$$

As presented in Sec. 3.2, these matrices are used within an iterative optimization method to refine an initial guess of the parameters towards a local minimum.

In order to perform the optimization, we also depend on the evaluation of the stacked vector of the residuals evaluated at the current estimate and left-multiplied by $\widetilde{\mathbf{\Sigma}}^{-\frac{1}{2}}$, *i.e.*,

$$\Delta\mathbf{T} \quad = \quad \begin{pmatrix} \widetilde{\mathbf{\Sigma}}_{\mathbf{X}_1}^{-\frac{1}{2}} \mathbf{e}_1 \\ \vdots \\ \widetilde{\mathbf{\Sigma}}_{\mathbf{X}_N}^{-\frac{1}{2}} \mathbf{e}_N \end{pmatrix}. \tag{3.57}$$

Following the argumentation in Sec. 2.2.2, we finally require the scaling matrices $\mathbf{G}_{\boldsymbol{\theta}}$ for $\mathbf{J}_{\boldsymbol{\theta}}$ and $\mathbf{G}_{\boldsymbol{\psi}}$ for $\mathbf{J}_{\boldsymbol{\psi}}$, defined as

$$\begin{aligned} \mathbf{G}_{\boldsymbol{\theta}} &= \operatorname{diag}\left[\frac{1}{\left\|\mathbf{j}_{1_{\boldsymbol{\theta}}}\right\|_2}, \ldots, \frac{1}{\left\|\mathbf{j}_{K_{\boldsymbol{\theta}}}\right\|_2} \right], \\ \mathbf{G}_{\boldsymbol{\psi}} &= \operatorname{diag}\left[\frac{1}{\left\|\mathbf{j}_{1_{\boldsymbol{\psi}}}\right\|_2}, \ldots, \frac{1}{\left\|\mathbf{j}_{L_{\boldsymbol{\psi}}}\right\|_2} \right], \end{aligned} \tag{3.58}$$

where the $\mathbf{j}_{i_{\boldsymbol{\theta}}}$ and $\mathbf{j}_{i_{\boldsymbol{\psi}}}$ are the column vectors of $\mathbf{J}_{\boldsymbol{\theta}}$ and $\mathbf{J}_{\boldsymbol{\psi}}$ respectively. The scaling matrices right-multiply their corresponding Jacobian matrix and the problem is solved for the scaled matrices. Once scaled solutions $\Delta\boldsymbol{\theta}_S$ and $\Delta\boldsymbol{\psi}_S$ are found, they can be left-multiplied to retrieve the unscaled solutions, *i.e.*,

$$\begin{aligned} \Delta\boldsymbol{\theta} &= \mathbf{G}_{\boldsymbol{\theta}} \Delta\boldsymbol{\theta}_S, \\ \Delta\boldsymbol{\psi} &= \mathbf{G}_{\boldsymbol{\psi}} \Delta\boldsymbol{\psi}_S. \end{aligned} \tag{3.59}$$

Obviously, if one of the column norms in Equ. 3.58 is zero, the corresponding entry in the scaling matrix will be zero. Nevertheless, one should be aware of numerical issues. Indeed, a column norm will never be exactly zero in practice, but might be very close to it. If no action is taken in that case, we might totally change the matrix structure after scaling. To cope with this issue, we recall the concept of *machine epsilon* or *unit roundoff*, henceforth denoted by u. The notion of unit roundoff is widely used by numerical analysts [Hig96] and provides an upper bound to the relative error occurring when performing floating-point operations on a computer subject to finite precision arithmetic. In our case, it might get complicated to perform the full error analysis since the entries in the Jacobian matrices are the results of prior computations which have already accumulated numerical errors. However, we can use a heuristic to define a bound for the vector norm $\|\mathbf{x}\|_2$ being zero, where $\mathbf{x} \in \mathbb{R}^n$,

$$\|\mathbf{x}\|_2 \quad = \quad \begin{cases} 0 & \text{for } \|\mathbf{x}\|_2 \leq nu \\ \|\mathbf{x}\|_2 & \text{otherwise.} \end{cases} \tag{3.60}$$

Alternatively, the unit roundoff u can be replaced by an estimate u_S of the expected roundoff error in the matrix entries.

3.4.2 Control Parameters

The sensor model being defined, our algorithm comprises several sets of adjustable parameters $\mathcal{P}_N = \{u_S, \epsilon_{\boldsymbol{\theta}}, \epsilon_{\boldsymbol{\psi}}\}$, $\mathcal{P}_I = \{\Delta N, \delta\}$, $\mathcal{P}_O = \{t_{max}, \varepsilon\}$, and $\mathcal{P}_M = \{p, w_q\}$. The set $\mathcal{P}_N$ is related to the numerical behavior, $\mathcal{P}_I$ to the online behavior, $\mathcal{P}_O$ to the nonlinear optimization, and $\mathcal{P}_M$ to the dampening of outliers.

The parameters $\epsilon_{\boldsymbol{\theta}}$ and $\epsilon_{\boldsymbol{\psi}}$ control the numerical rank determination of $\mathbf{A}_{\boldsymbol{\theta}}$ from Equ. 3.17 and $\mathbf{J}_{\boldsymbol{\psi}}$ respectively. We have already elaborated on this topic in Sec. 2.2.2. If we knew the exact perturbation matrices $\mathbf{E}_{\boldsymbol{\theta}}$ and $\mathbf{E}_{\boldsymbol{\psi}}$ applied to the noiseless matrices, we would be able to set these thresholds accordingly. However, in our application, these perturbation matrices are not straightforward to calculate. They are composed of roundoff errors and of linearization errors in nonlinear least-squares problems. In that case, the unperturbed matrices would correspond to the Jacobian matrices evaluated at the true value of $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$. The perturbation matrices are thus essentially dominated by the discrepancy between the current estimate and the true value of the parameters. We shall demonstrate in Chap. 4 that these local estimates can lead to updates in the unobservable subspace if no proper action is engaged. In order to come up with operational bounds, we shall first state a generic heuristic for a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ containing only roundoff errors

$$\epsilon_{\mathbf{A}} = f(m,n) u \left\| \mathbf{A} \right\|_p , \tag{3.61}$$

where u is again the unit roundoff and $f(\cdot)$ is a slowly growing function of m and n. The matrix p-norm is usually the 2-norm, the 1-norm, the ∞-norm, or the Frobenius norm, depending on the applications. In our setup, we can adapt this heuristic in the following manner

$$\begin{aligned} \epsilon_{\boldsymbol{\theta}} &= K^2 u_{\boldsymbol{\theta}} \left\| \mathbf{A}_{\boldsymbol{\theta}} \right\|_2 , \\ \epsilon_{\boldsymbol{\psi}} &= (M+L) u_{\boldsymbol{\psi}} \left\| \mathbf{J}_{\boldsymbol{\psi}} \right\|_\infty , \end{aligned} \tag{3.62}$$

where $u_{\boldsymbol{\theta}}$ and $u_{\boldsymbol{\psi}}$ are upper bounds for the noise in the entries of $\mathbf{A}_{\boldsymbol{\theta}}$ and $\mathbf{J}_{\boldsymbol{\psi}}$ respectively. These values are application specific and we leave them as free parameters for our algorithm. We shall investigate more on that topic in Chap. 4. The 2-norm is already available from the SVD in Equ. 3.20 and the ∞-norm is the maximum absolute row sum. Given these bounds, we update the set of numerical parameters to $\mathcal{P}_N = \{u_S, u_{\boldsymbol{\theta}}, u_{\boldsymbol{\psi}}\}$.

The parameters ΔN and δ govern the online behavior of our algorithm. While ΔN controls the size of the batches evaluated, δ is a threshold for the information provided by a batch with respect to an existing marginal posterior for $\boldsymbol{\Theta}$. ΔN can represent a certain number of measurements or a time window during which measurements are gathered. This latter interpretation is particularly valuable in the presence of multiple asynchronous sensors. A large value for ΔN corresponds in the limit to a full batch method and is therefore not recommended for online operations. A small value for ΔN involves many unnecessary optimization runs and might not provide enough information to render the calibration parameters observable. The selection of ΔN is thus purely empirical, dependent on the vehicle dynamics, and is a trade-off between

online and offline behavior. The information threshold δ is given in bits. If it is set to a low value, most of the batches will be accepted and our algorithm will become largely inefficient as compared to a full batch method. On the other hand, when setting δ to a large value, we will discard most of the batches and fail to gather important information. As for ΔN, the choice of δ is empirical and application specific. These two parameters are furthermore related in the sense that larger batches can potentially reduce the uncertainty of the posterior distribution in a more significant way. We shall also examine the influence of these parameters in Chap. 4.

The remaining parameters t_{max} and ε control the convergence of the nonlinear optimization method. t_{max} is an upper bound for the number of iterations and ε a threshold for the relative change in the sum of the squared residuals or for the shift vector norm.

3.4.3 Algorithm and Complexity Analysis

Our calibration method is an online algorithm that continuously analyzes a stream of measurements. In that regard, we state the high-level update step for a new batch of data in Alg. 3.1. The intermediary steps are outlined in Alg. 3.2 and Alg. 3.4. The nonlinear least-squares algorithm can further be detailed for the shift vector computation in Alg. 3.3. Although stated for a single sensor $\mathcal{S}$, our algorithm generalizes well to multiple asynchronous sensors. Furthermore, the nonlinear least-squares part can be replaced by a linear least-squares solver if needed. The algorithm should be started with initial guesses for $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$, and requires the forward model $h_{\mathcal{S}}(\cdot)$, the analytical Jacobian matrices $\mathbf{J}(\boldsymbol{\theta})$ and $\mathbf{J}(\boldsymbol{\psi})$, the Cholesky factor of the inverse data covariance matrix $\boldsymbol{\Sigma}_{\mathbf{X}}^{-\frac{1}{2}}$, and the set of control parameters $\mathcal{P}$. If necessary, numerical Jacobian matrices can also be calculated iteratively in Alg. 3.2. Some function calls in the algorithms are not detailed, but should be straightforward to implement. As a picture is worth a thousand words, Fig. 3.1 provides a high-level view on the algorithmic flow.

Algorithm 3.1: `UpdateCalibration`$(\mathcal{D}^{new}, \mathcal{S}^{prior}, \mathcal{P})$

Input: New batch $\mathcal{D}^{new}$
Input: Current calibration state
$\mathcal{S}^{prior} = \{\boldsymbol{\Theta} \mid \mathcal{D}^{info}, \mathbf{R}_{\boldsymbol{\theta}}, \mathbf{N}_{\boldsymbol{\theta}}, \boldsymbol{\mu}_{\boldsymbol{\Psi}}, h_{\mathcal{S}}(\cdot), \mathbf{J}(\boldsymbol{\theta}), \mathbf{J}(\boldsymbol{\psi}), \boldsymbol{\Sigma}_{\mathbf{X}}^{-\frac{1}{2}}\}$
Input: Control parameters $\mathcal{P} = \{\mathcal{P}_N, \mathcal{P}_I, \mathcal{P}_O, \mathcal{P}_M\}$
Output: Updated calibration state $\mathcal{S}^{post}$

`// Nonlinear least-squares from Alg. 3.2:`
$\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}}, \boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}}, \mathbf{V}, r_{\epsilon_{\boldsymbol{\theta}}} \leftarrow$ `NonlinearLeastSquares`$(\mathcal{D}^{new}, \mathcal{S}^{prior}, \mathcal{P}_N, \mathcal{P}_O, \mathcal{P}_M)$

`// Information selection from Alg. 3.4:`
$\mathcal{S}^{post} \leftarrow$ `SelectInformation`$(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}}, \boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}}, \mathbf{V}, r_{\epsilon_{\boldsymbol{\theta}}}, \mathcal{S}^{prior}, \mathcal{P}_I)$

return $\mathcal{S}^{post}$

The complexity of our algorithm is dominated by the computations in Alg. 3.3. It involves matrix multiplications, rank-revealing QR decomposition, and singular value decomposition. Numerical analysts quantify arithmetic complexity

Algorithm 3.2: `NonlinearLeastSquares`$(\mathcal{D}^{new}, \mathcal{S}^{prior}, \mathcal{P}_N, \mathcal{P}_O, \mathcal{P}_M)$

Input: New batch $\mathcal{D}^{new}$
Input: Current calibration state $\mathcal{S}^{prior}$
Input: Control parameters $\mathcal{P}_N, \mathcal{P}_O, \mathcal{P}_M$
Output: Best estimates $\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}}, \boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}}$
Output: Right singular vectors $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_K)$ and numerical rank $r_{\epsilon_{\boldsymbol{\theta}}}$

`// Initializations:`
$\widehat{\boldsymbol{\theta}}_{(0)}, \widehat{\boldsymbol{\psi}}_{(0)} \leftarrow \boldsymbol{\mu}_{\boldsymbol{\Theta}}, \boldsymbol{\mu}_{\boldsymbol{\Psi}}$
$t \leftarrow 1$
converged $\leftarrow$ **false**

`// Nonlinear optimization loop:`
while not *converged* **do**
  $\mathbf{J}_{\boldsymbol{\theta}}, \mathbf{J}_{\boldsymbol{\psi}} \leftarrow$ `EvaluateJacobians`$(\widehat{\boldsymbol{\theta}}_{(t-1)}, \widehat{\boldsymbol{\psi}}_{(t-1)}, \mathcal{D}^{new}, \mathcal{S}^{prior}, \mathcal{P}_M)$
  $\Delta\mathbf{T} \leftarrow$ `EvaluateError`$(\widehat{\boldsymbol{\theta}}_{(t-1)}, \widehat{\boldsymbol{\psi}}_{(t-1)}, \mathcal{D}^{new}, \mathcal{S}^{prior}, \mathcal{P}_M)$
  $\mathbf{G}_{\boldsymbol{\theta}}, \mathbf{G}_{\psi} \leftarrow$ `ScalingMatrix` $(\mathbf{J}_{\boldsymbol{\theta}}, \mathbf{J}_{\psi})$ `// See Equ. 3.58`
  $\Delta\boldsymbol{\theta}_S, \Delta\boldsymbol{\psi}_S, \mathbf{V}, r_{\epsilon_{\boldsymbol{\theta}}} \leftarrow$ `ComputeShiftVector`$(\mathbf{J}_{\boldsymbol{\theta}}\mathbf{G}_{\boldsymbol{\theta}}, \mathbf{J}_{\boldsymbol{\psi}}\mathbf{G}_{\psi}, \Delta\mathbf{T}, \mathcal{P}_N)$
  $\widehat{\boldsymbol{\theta}}_{(t)} \leftarrow \widehat{\boldsymbol{\theta}}_{(t-1)} + \mathbf{G}_{\boldsymbol{\theta}}\Delta\boldsymbol{\theta}_S$
  $\widehat{\boldsymbol{\psi}}_{(t)} \leftarrow \widehat{\boldsymbol{\psi}}_{(t-1)} + \mathbf{G}_{\psi}\Delta\boldsymbol{\psi}_S$
  $t \leftarrow t + 1$
  converged $\leftarrow$ `ConvergenceCheck`$(t, \Delta\boldsymbol{\theta}, \Delta\boldsymbol{\psi}, \Delta\mathbf{T}, \mathcal{P}_O)$
end

`// Best estimates:`
$\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}} \leftarrow \widehat{\boldsymbol{\theta}}_{(t)}, \widehat{\boldsymbol{\psi}}_{(t)}$

`// Covariance at the mode:`
$\mathbf{J}_{\boldsymbol{\theta}}, \mathbf{J}_{\boldsymbol{\psi}} \leftarrow$ `EvaluateJacobians`$(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}}, \mathcal{D}^{new}, \mathcal{S}^{prior}, \mathcal{P}_M)$
$\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}} \leftarrow$ `ComputeCovariance`$(\mathbf{J}_{\boldsymbol{\theta}}, \mathbf{J}_{\boldsymbol{\psi}}, r_{\epsilon_{\boldsymbol{\theta}}})$ `// See Equ. 3.22`

return $\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}}, \boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}}, \mathbf{V}, r_{\epsilon_{\boldsymbol{\theta}}}$

Algorithm 3.3: `ComputeShiftVector`$(\mathbf{J}_{\boldsymbol{\theta}}, \mathbf{J}_{\boldsymbol{\psi}}, \Delta\mathbf{T}, \mathcal{P}_N)$

Input: Jacobian matrices $\mathbf{J}_{\boldsymbol{\theta}}, \mathbf{J}_{\boldsymbol{\psi}}$ and error vector $\Delta\mathbf{T}$
Input: Control parameters $\mathcal{P}_N$
Output: Shift vectors $\Delta\boldsymbol{\theta}$ and $\Delta\boldsymbol{\psi}$
Output: Right singular vectors $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_K)$ and numerical rank $r_{\epsilon_{\boldsymbol{\theta}}}$

$\mathbf{Q}_1, \mathbf{Q}_{11}, \mathbf{R}_{11}, \boldsymbol{\Pi}, r_{\epsilon_{\boldsymbol{\psi}}} \leftarrow$ `RRQR` $(\mathbf{J}_{\psi}, \epsilon_{\boldsymbol{\psi}})$ `// See Equ. 3.15`
$\mathbf{A}_{\boldsymbol{\theta}} \leftarrow \mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{J}_{\boldsymbol{\theta}} - (\mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1)(\mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1)^T$
$\mathbf{b}_{\boldsymbol{\theta}} \leftarrow \mathbf{J}_{\boldsymbol{\theta}}^T\Delta\mathbf{T} - (\mathbf{J}_{\boldsymbol{\theta}}^T\mathbf{Q}_1)(\Delta\mathbf{T}^T\mathbf{Q}_1)^T$
$\mathbf{U}, \mathbf{S}, \mathbf{V}, r_{\epsilon_{\boldsymbol{\theta}}} \leftarrow$ `SVD`$(\mathbf{A}_{\boldsymbol{\theta}}, \epsilon_{\boldsymbol{\theta}})$ `// See Equ. 3.20`
$\Delta\boldsymbol{\theta} \leftarrow$ `TSVD`$(\mathbf{U}, \mathbf{S}, \mathbf{V}, \mathbf{b}_{\boldsymbol{\theta}}, r_{\epsilon_{\boldsymbol{\theta}}})$ `// See Equ. 3.21`
$\Delta\boldsymbol{\psi} \leftarrow$ `BasicSolution`$(\boldsymbol{\Pi}, \mathbf{R}_{11}, \mathbf{Q}_{11}, \Delta\mathbf{T}, \mathbf{J}_{\boldsymbol{\theta}}, \Delta\boldsymbol{\theta}, r_{\epsilon_{\boldsymbol{\psi}}})$ `// See Equ. 3.27`

return $\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\psi}, \mathbf{V}, r_{\epsilon_{\boldsymbol{\theta}}}$

Algorithm 3.4: `SelectInformation`$(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}}, \boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}}, \mathbf{V}, r_{\epsilon_{\boldsymbol{\theta}}}, \mathcal{S}^{prior}, \mathcal{P}_I)$

Input: Best estimates $\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\psi}}, \boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}}$
Input: Right singular vectors $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_K)$ and numerical rank $r_{\epsilon_{\boldsymbol{\theta}}}$
Input: Current calibration state $\mathcal{S}^{prior}$
Input: Control parameters $\mathcal{P}_I$
Output: Updated calibration state $\mathcal{S}^{post}$

$\boldsymbol{\Theta} \mid \mathcal{D}^{info}, \mathcal{D}^{new} \leftarrow \mathcal{N}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\widehat{\boldsymbol{\theta}}})$

if $\mathcal{U}((\boldsymbol{\Theta} \mid \mathcal{D}^{info}), \mathcal{D}^{new}) > \delta$ **then**
  $\mathcal{D}^{info} \leftarrow \mathcal{D}^{info} \cup \mathcal{D}^{new}$
  $\boldsymbol{\Theta} \mid \mathcal{D}^{info} \leftarrow \boldsymbol{\Theta} \mid \mathcal{D}^{info}, \mathcal{D}^{new}$
  $\mathbf{R}_{\boldsymbol{\theta}} \leftarrow (\mathbf{v}_1, \ldots, \mathbf{v}_{r_{\epsilon_{\boldsymbol{\theta}}}})$
  $\mathbf{N}_{\boldsymbol{\theta}} \leftarrow (\mathbf{v}_{r_{\epsilon_{\boldsymbol{\theta}}}+1}, \mathbf{v}_K)$
  $\boldsymbol{\mu}_{\boldsymbol{\Psi}} \leftarrow \widehat{\boldsymbol{\psi}}$
else
  $\mathcal{S}^{post} \leftarrow \mathcal{S}^{prior}$
end

return $\mathcal{S}^{post}$

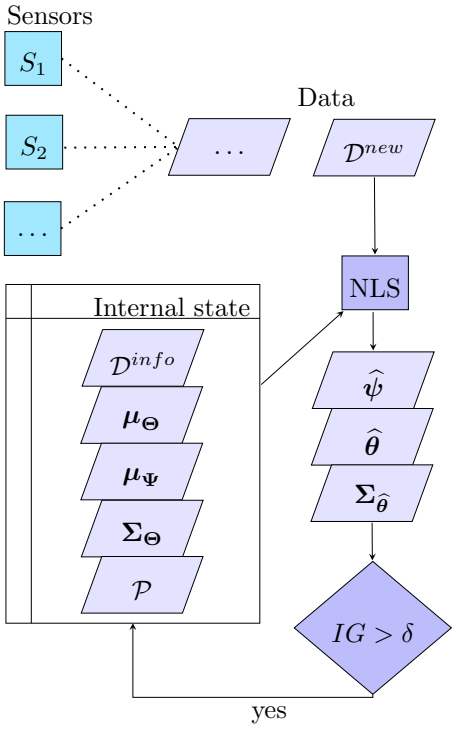


Figure 3.1: Algorithmic flow. Sensor data are collected into batches $\mathcal{D}^{new}$ that are sequentially used, along with $\mathcal{D}^{info}$, to estimate a tentative refinement of the latent variables $\boldsymbol{\Psi}$ and $\boldsymbol{\Theta}$. Upon termination of the nonlinear least-squares estimator, an information gain test decides whether to update the internal state or discard the newly computed estimate.

with *flop counts*. The term *flop* is an abbreviation for floating-point operation, *i.e.*, any arithmetic operation on floating-point numbers. The flop count for matrix decompositions largely depends on the implementation. As reported in [GL96], a standard SVD requires $4m^2n + 22n^3$ flops for an $m \times n$ matrix and an RRQR $4mnr - 2r^2(m+n) + 4r^3/3$ flops for a rank-r matrix. General matrix multiplication between an $m \times p$ and a $p \times n$ matrix involves $2mnp$ flops. Using these figures, we can accurately estimate the flop count of Alg. 3.3 for $\mathbf{J}_{\boldsymbol{\theta}} \in \mathbb{R}^{M \times K}$ and $\mathbf{J}_{\boldsymbol{\psi}} \in \mathbb{R}^{M \times L}$. However, we believe the *big-Oh notation* [Gol08] is more appropriate to express the trend of the algorithm with respect to its inputs. The big-Oh notation describes the limiting behavior, either running time or memory storage, of an algorithm as a function of the size of its inputs, usually in terms of simpler functions than the flop count. We can assume that $M \geq (L + K)$ and in the worst case $\operatorname{rank}[\mathbf{J}_{\boldsymbol{\psi}}] = L$. The total flop count of Alg. 3.3 is a complex function of M, L, and K. If L and K are kept constant, the running time of the algorithm is $\mathcal{O}(M)$, *i.e.*, growing linearly when adding new measurements. In our application, however, we can regard the dimension of the calibration variable as constant, but L is typically growing with M. As stated in Sec. 3.1, $\boldsymbol{\Psi}$ might represent poses of landmarks, which number might, in the worst case, grow with each measurement. In that scenario, we have a running-time complexity of $\mathcal{O}(L^3)$ or $\mathcal{O}(M^3)$, which can quickly become prohibitive for a reasonable dataset (*e.g.*, several minutes of computation on a cutting-edge multi-core processor). From the consistency of the underlying estimator, we can nevertheless assume that $\boldsymbol{\Theta}$ converges to a Dirac distribution given an infinite amount of measurements. Therefore, our information selection scheme in Alg. 3.4 bounds the *time complexity* as compared to an offline method. The memory cost is dominated by the storage of the Jacobian matrices, *i.e.*, which leads in the worst case to a *space complexity* of $\mathcal{O}(M^2)$ or $\mathcal{O}(L^2)$. For typical datasets, this can represent several gigabytes of RAM. Here also, we can nonetheless safely assume that the space complexity is bounded by our online algorithm.

Thus far, we have presented the basic principles of our algorithm, but have not exploited the structure of the probabilistic model. As praised by Bishop [Bis06], a *probabilistic graphical model* is a valuable tool to visualize the complexity of a probabilistic model. In most applications, we can utilize several conditional independence assumptions such that the joint data distribution is factorized into a product of elementary densities. Intuitively, if $\mathbf{X}_i$ represents the measurement of the pose of a landmark, its value is independent from the pose of other landmarks. We can illustrate this with the *Bayesian network* in Fig. 3.2 representing a joint distribution of a data sample $\mathbf{X}_1, \ldots, \mathbf{X}_N$, the calibration variable $\boldsymbol{\Theta} = (\Theta_1, \ldots, \Theta_K)^T$, and the nuisance variable $\boldsymbol{\Psi} = (\Psi_1, \ldots, \Psi_L)^T$. From Fig. 3.2, we witness that the joint distribution can be factorized into a product of local conditional distributions. As a consequence, the Jacobian matrices $\mathbf{J}_{\boldsymbol{\theta}}$ and $\mathbf{J}_{\boldsymbol{\psi}}$ will contain a *sparsity pattern*, *i.e.*, each conditional distributions contributes to the Jacobian matrices by adding a small block of nonzero elements. If the Bayesian network contains few arrows with respect to the number of variables, the Jacobian matrices can thus be stored efficiently in memory using a sparse matrix data structure [Tew76], *e.g.*, *compressed row storage (CRS)* or *compressed column storage (CCS)*. If we denote the number of nonzero elements in the Jacobian matrices by nnz, with nnz being $\mathcal{O}(L)$, the space complexity becomes in the worst case $\mathcal{O}(L)$, which is much more affordable. The

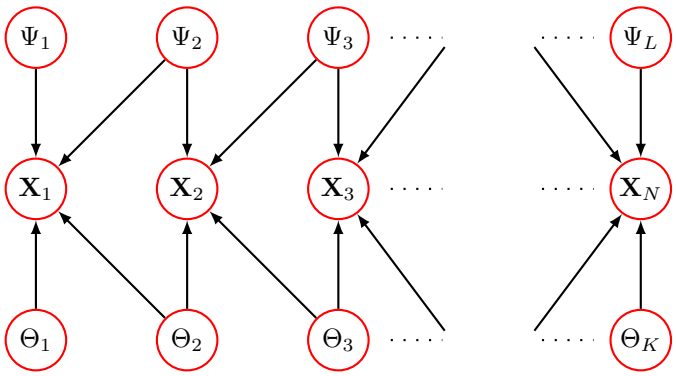


Figure 3.2: The Bayesian network representation of a factorization for the joint probability density function $f(\mathbf{X}_1, \ldots, \mathbf{X}_N, \boldsymbol{\Theta}, \boldsymbol{\Psi})$.

CCS or CRS format also improves the time complexity of matrix operations. For instance, matrix multiplication between an $m \times p$ and a $p \times n$ matrix can be implemented in $\mathcal{O}(n)$. The QR decomposition in Alg. 3.3 can also exploit the sparsity pattern [Dav11]. A sparse QR decomposition consists in two distinct phases, namely a symbolic and a numerical phase. The symbolic phase determines the sparsity pattern of the $\mathbf{R}$ matrix and a permutation reducing the fill-in. Although fill-reducing ordering is theoretically *NP-complete* [Yan81], there exists heuristics in practice to reach $\mathcal{O}(L)$ [Dav06]. Furthermore, the symbolic factorization can be reused for each iteration of the optimization algorithm. The second phase performs the numerical factorization using specialized orthogonalization techniques. As reported in [GN88], we can hope for a time complexity in $\mathcal{O}(L^{\frac{3}{2}})$. In summary, if we assume the dimension K of the calibration variable $\boldsymbol{\Theta}$ small and constant, $nnz(\mathbf{J}_{\boldsymbol{\psi}})$ in $\mathcal{O}(L)$ with L the dimension of the nuisance variable $\boldsymbol{\Psi}$, the overall time complexity of Alg. 3.1 is $\mathcal{O}(L^{\frac{3}{2}})$ and the space complexity $\mathcal{O}(L)$.

We have seen that when the Bayesian network is sparse, the sparsity of the Jacobian matrices can be beneficially exploited. Another important insight from the graphical model theory is that the computations are essentially local. If we transform Fig. 3.2 into a *Markov random field (MRF)*, the calculations can be performed separately for each clique. Therefore, our algorithm can be parallelized to reduce the cost of Jacobian and error term evaluations. Along these lines, the sparse QR decomposition in [Dav11] also offers opportunities for parallel computations. As a final remark, if the potential functions for the cliques are expressed in terms of energy functions, our MAP estimation framework can be formulated as an energy minimization problem [SZS+08].

4 Applications and Experiments

If we begin with certainties, we shall end in doubts; but if we begin with doubts, and are patient in them, we shall end in certainties.

Sir Francis Bacon (1561-1626)

In this chapter, we shall demonstrate three concrete applications for our calibration algorithm presented in Chap. 3. We start with a standard problem often encountered in undergraduate robotic lectures, namely a differential-drive wheeled robot observing landmarks with a laser rangefinder. Increasing the difficulty, we move to the three-dimensional space for a more realistic application involving multiple asynchronous sensors in a consumer car. Lastly, we study the typical computer vision problem which consists in determining the intrinsic calibration parameters of a camera. Besides a complete description of the application of our algorithm to these problems, we provide a thorough experimental evaluation with simulated and real-world data. As a preliminary basis, we establish a couple of conventions and notations to represent the position, orientation, and motion of objects in space.

4.1 Conventions and Notations

In Sec. 2.2.1, we have discussed the concept of vector space in an abstract manner. We shall here focus on the *Euclidean space*, *i.e.*, a vector space $\mathbb{R}^n$ over the real field $\mathbb{R}$ equipped with an *inner product*. This mathematical structure provides the concepts of vector lengths and angles between vectors.

4.1.1 Affine Space

Along with the vector space $\mathbb{R}^n$, we can define an *affine space* $\mathcal{P}$, whose elements are called *points*. For two points $p_1, p_2 \in \mathcal{P}$, there is a unique vector $\mathbf{r}^{p_1p_2} \in \mathbb{R}^n$ such that $\mathbf{r}^{p_1p_2} = p_2 - p_1$. Furthermore, the addition of the vector $\mathbf{r}^{p_1p_2}$ to a point p_1, referred to as *translation*, yields another point $p_3 \in \mathcal{P}$ such that $p_3 = \mathbf{r}^{p_1p_2} + p_1$.

If we define an orthonormal basis $\{\mathbf{e}_1, \dots, \mathbf{e}_n\}$ in the affine space $(\mathcal{P}, \mathbb{R}^n)$, the basis vectors, referred to as *axes*, intersect at a point known as the *origin*. Altogether, this describes a *Cartesian coordinate system* that we denote by $\mathcal{F} = \{o, \mathbf{e}_1, \dots, \mathbf{e}_n\}$. We shall henceforth use the term *coordinate frame* or simply *frame* to refer to a Cartesian coordinate system. In order to distinguish between different coordinate frames, we label them with a subscript notation as $\mathcal{F}_A$. When required, the point of origin may also be labeled as o_A or simply a.

For any point $p \in \mathcal{P}$ and the origin point $a \in \mathcal{P}$ of $\mathcal{F}_A$, we define a *position vector* $\mathbf{r}^{ap} \in \mathbb{R}^n$. Recalling the concepts in Sec. 2.2.1, the vector $\mathbf{r}^{ap}$ can be expressed as a linear combination of the basis vectors. If we consider an ordered basis, the vector is uniquely identified by the scalar coefficients, referred to as *coordinates*, of the linear combination. We use these coordinates to describe

the point p uniquely with respect to $\mathcal{F}_A = \{a, \mathbf{e}_1, \ldots, \mathbf{e}_n\}$ as a *column vector* ${}^a\mathbf{p} = ({}^a p_1, \ldots, {}^a p_n)^T$, where ${}^a\mathbf{r}^{ap} = {}^a p_1 \mathbf{e}_1 + \ldots + {}^a p_n \mathbf{e}_n$. When required, we use a similar representation for a position vector, *i.e.*, ${}^a\mathbf{r}^{ap} = ({}^a p_1, \ldots, {}^a p_n)^T$.

4.1.2 Pose Representation

A *rigid body* is constituted of a set of points whose relative distance remains constant with respect to an attached coordinate frame. The *pose* of the rigid body comprises the *position* and *orientation* of its coordinate frame. In our thesis, we shall be interested in the relative pose between a body's coordinate frame and a reference frame. We shall further distinguish between different types of frames, the *world frame* $\mathcal{F}_W$, *robot frames* $\mathcal{F}_{R_i}$ or *vehicle frames* $\mathcal{F}_{V_i}$, and *sensor frames* $\mathcal{F}_{S_i}$. The world frame corresponds to an *inertial reference frame* from which the motion of the rigid bodies is observed. The extrinsic calibration problem boils down to estimating the relative pose, assumed constant over time, between a sensor and a vehicle frame. As a byproduct, we shall also require the relative pose between the world and the vehicle frame.

The relative position between the coordinate frames $\mathcal{F}_A$ with origin a and $\mathcal{F}_B$ with origin b can be trivially represented by the translation vector ${}^a\mathbf{r}^{ab} \in \mathbb{R}^n$. Although the frame origin of a rigid body can be chosen arbitrarily, it commonly coincides with its center of mass.

The relative orientation between two frames can be described by a *rotation*. A rotation maps an orthonormal basis in $\mathbb{R}^n$ to another orthonormal basis in $\mathbb{R}^n$. As a rotation is a linear map, it can be represented by an $n \times n$ *rotation matrix*, *i.e.*, an orthogonal matrix with unitary determinant. The set of orthogonal matrices representing proper rotations forms a special orthogonal group known as $SO(n)$. Unlike translation, the concept of rotation has multiple interpretations and representations, which can lead to ambiguities. For the purpose of this thesis, we consider a rotation as the change of viewpoint under which a vector is observed. For a vector $\mathbf{v} \in \mathbb{R}^n$ expressed as a column vector of coordinates ${}^b\mathbf{v}$ with respect to a frame $\mathcal{F}_B$, its coordinates ${}^a\mathbf{v}$ with respect to the frame $\mathcal{F}_A$ are given by

$$ {}^a\mathbf{v} \quad = \quad {}^a\mathbf{R}_b \, {}^b\mathbf{v}, \tag{4.1} $$

where ${}^a\mathbf{R}_b \in \mathbb{R}^{n \times n}$ is a rotation matrix. The inverse rotation matrix ${}^b\mathbf{R}_a$ can be computed by transposing ${}^a\mathbf{R}_b$. If we define another frame $\mathcal{F}_C$, a rotation matrix ${}^b\mathbf{R}_c$, and a vector ${}^c\mathbf{v}$, rotations can be composed to give

$$ {}^a\mathbf{v} \quad = \quad {}^a\mathbf{R}_b \, {}^b\mathbf{R}_c \, {}^c\mathbf{v}. \tag{4.2} $$

In an affine space, translation and rotation can be combined into an *affine transformation*, that can be represented by a *transformation matrix* of the form

$$ {}^a\mathbf{T}_b \quad = \quad \begin{pmatrix} {}^a\mathbf{R}_b & {}^a\mathbf{r}^{ab} \\ \mathbf{0} & 1 \end{pmatrix}. \tag{4.3} $$

The set of transformation matrices in $\mathbb{R}^n$ form a special Euclidean group known as $SE(n)$. The transformation matrix is a compact representation of the relative

pose between two frames; it contains the translation vector between the origin points and the rotation matrix for the basis vectors. For a point ${}^{b}\mathbf{p}$, we can compute its coordinates ${}^{a}\mathbf{p}$ in the frame $\mathcal{F}_A$ using the following relation

$$ {}^{a}\widetilde{\mathbf{p}} \quad = \quad {}^{a}\mathbf{T}_b \, {}^{b}\widetilde{\mathbf{p}}, \tag{4.4} $$

where

$$ {}^{b}\widetilde{\mathbf{p}} \quad = \quad \begin{pmatrix} {}^{b}\mathbf{p} \\ 1 \end{pmatrix} \tag{4.5} $$

is an augmented column vector, whose components are henceforth referred to as *homogeneous coordinates*. An affine transformation therefore becomes a linear map when using homogeneous coordinates. As for rotations, affine transformations can be composed by multiplying their transformation matrices. The inverse affine transformation ${}^{b}\mathbf{T}_a$ can be computed by inverting the matrix ${}^{a}\mathbf{T}_b$, or is simply given by

$$ {}^{b}\mathbf{T}_a \quad = \quad \begin{pmatrix} {}^{a}\mathbf{R}_b^T & -{}^{a}\mathbf{R}_b^T \, {}^{a}\mathbf{r}^{ab} \\ \mathbf{0} & 1 \end{pmatrix}. \tag{4.6} $$

To keep the notation uncluttered, we assume from now on that transformation matrices representing affine transformations always apply to points in homogeneous coordinates. For affine transformations, the homogeneous component is never altered and we might therefore simply omit it or add it to match the dimensional context.

4.1.3 Motion Representation

If a rigid body undergoes motion with respect to an inertial reference frame, its relative pose is varying with time. As for the pose, the motion has a linear and an angular component. If we denote the inertial reference frame as $\mathcal{F}_W$ and a body frame as $\mathcal{F}_V$, the translation vector ${}^{w}\mathbf{r}^{wv}$ is a time-varying quantity that we express as ${}^{w}\mathbf{r}^{wv}(t)$. The *linear velocity* of $\mathcal{F}_V$ with respect to $\mathcal{F}_W$ is defined as the time derivative of ${}^{w}\mathbf{r}^{wv}$, *i.e.*,

$$ {}^{w}\mathbf{v}^{wv}(t) \quad = \quad {}^{w}\dot{\mathbf{r}}^{wv}(t). \tag{4.7} $$

The notation ${}^{w}\mathbf{v}^{wv}(t)$ signifies the linear velocity of the origin point of $\mathcal{F}_V$ observed in $\mathcal{F}_W$ and expressed in $\mathcal{F}_W$.

The *angular velocity* describes the rate at which a rigid body is rotating around its rotation axis. Indeed, a rotation can always be represented as a planar rotation of an angle $\varphi(t)$ around an instantaneous axis of rotation ${}^{w}\mathbf{u}(t)$. Therefore, the angular velocity vector can be defined as

$$ {}^{w}\boldsymbol{\omega}^{wv}(t) \quad = \quad \dot{\varphi}(t) \, {}^{w}\mathbf{u}(t). \tag{4.8} $$

The angular velocity can also be derived in term of the rotation matrix as

$$[{}^{w}\boldsymbol{\omega}^{wv}]_{\times}(t) \;=\; {}^{w}\dot{\mathbf{R}}_{v}(t)\,{}^{w}\mathbf{R}_{v}^{T}(t), \tag{4.9}$$

where the *skew-symmetric matrix* $[{}^{w}\boldsymbol{\omega}^{wv}]_{\times}(t)$ represents the *angular velocity tensor*.

All the points in $\mathcal{F}_V$ have the same angular velocity. On the other hand, the linear velocity of the points increases with the distance to the instantaneous axis of rotation. If we denote the origin of $\mathcal{F}_V$ by v, a point on the body by p, and the vector from v to p by ${}^{w}\mathbf{r}^{vp}$, we can compute the linear velocity of p observed from $\mathcal{F}_W$ as

$${}^{w}\mathbf{v}^{wp}(t) \;=\; {}^{w}\mathbf{v}^{wv}(t) \;+\; {}^{w}\boldsymbol{\omega}^{wp}(t) \;\times\; {}^{w}\mathbf{r}^{vp}, \tag{4.10}$$

where $\times$ denotes the *cross product*.

4.2 Laser Rangefinder on a Differential-Drive Robot

In this first application, we consider a *differential-drive wheeled robot* equipped with a *laser rangefinder (LRF)*. In that situation, the LRF is referred to as an *exteroceptive sensor*. While moving in a plane, the LRF measures distances to poles disseminated on the ground. Furthermore, the robot is endowed with *interoceptive sensors*, *e.g.*, *odometry sensors*, measuring its linear and angular velocity. This scenario, displayed in Fig. 4.1, is prototypical for a *simultaneous localization and mapping (SLAM)* problem [DWB06, BDW06]. Freely adopting the SLAM jargon, we refer to the poles as *landmarks*. The positions in an inertial reference frame of a collection of landmarks form a *map* of the environment. Given an initial inertial reference frame, the SLAM problem consists in estimating the poses of the robot, *i.e.*, its trajectory, and the positions of the landmarks, solely based on the interoceptive and exteroceptive sensors. In order to solve this problem, we require the relative pose between the LRF frame and the robot frame, and the relative pose between the robot and the inertial frame. The LRF measurements shall indeed be transformed in the inertial frame to construct a consistent map. Recalling our presentation in Chap. 3, the relative pose between the LRF frame and the robot frame corresponds to the calibration variable, and the relative poses between the robot and the inertial frame, along with the positions of the landmarks, to the nuisance variable.

This scenario was first showcased in [MFS13] with a preliminary version of our algorithm. In the rest of this section, we shall phrase the underlying mathematical model and explore the problem to its full extent. Due to its simplicity, we can thoroughly analyze the behavior of our algorithm with extensive simulations and real-world experiments.

4.2.1 Problem Formulation

Before starting, we shall first identify the different coordinate frames in the problem. The inertial reference frame or world frame is represented by $\mathcal{F}_W$ with origin w. As we only compute relative poses to $\mathcal{F}_W$, its absolute pose is arbitrary. In this application, we collocate it with the *standard basis* in $\mathbb{R}^2$ and set the origin to ${}^{sb}\mathbf{w} = (0,0)^T$. The robot frame $\mathcal{F}_R$ has its origin r located at

Figure 4.1: Illustration of the setup for the differential-drive robot example (image courtesy of Timothy D. Barfoot, university of Toronto, 2009). While observing landmarks with a laser rangefinder and measuring its linear and angular velocity with odometry sensors, the robot builds a map of the environment in which it simultaneously localizes. In this scenario, the calibration problem consists in determining the relative pose between the laser rangefinder and robot frames.

the center of the axle between the left and right wheels. The first axis of $\mathcal{F}_R$ is aligned with the direction of forward motion of the robot and the second axis points to the left. The LRF frame $\mathcal{F}_L$ has the origin l at the point of emission of the laser beams. The first axis of $\mathcal{F}_L$ points along the forward direction of the LRF and the second axis to the left.

The relative pose of $\mathcal{F}_R$ with respect to $\mathcal{F}_W$ is a time-varying transformation denoted by ${}^{w}\mathbf{T}_r(t) \in SE(2)$, composed of a rotation ${}^{w}\mathbf{R}_r(t) \in SO(2)$ and a translation ${}^{w}\mathbf{r}^{wr}(t) \in \mathbb{R}^2$. The relative pose of $\mathcal{F}_L$ with respect to $\mathcal{F}_R$ is a time-invariant transformation denoted by ${}^{r}\mathbf{T}_l \in SE(2)$, composed of a rotation ${}^{r}\mathbf{R}_l \in SO(2)$ and a translation ${}^{r}\mathbf{r}^{rl} \in \mathbb{R}^2$. The setup of the scene is summarized in Fig. 4.2.

Observation Model

The playground of the robot contains a set of Nl landmarks represented by the points $\{\ell_1, \ldots, \ell_{Nl}\}$. We assume that the LRF returns for each landmark ℓ_i a *range* r^{ℓ_i} and a *bearing angle* φ^{ℓ_i}. If a landmark is not in the field of view of the LRF, its corresponding range is set to zero and flagged as invalid. Despite being a rather simplified setup, it is perfectly sufficient for demonstrating our approach, while circumventing the problem of landmark detection and association between consecutive laser scans. The range r^{ℓ_i} corresponds to the distance between the origin l and the landmark ℓ_i. In other words, it is the 2-norm of the vector ${}^{l}\mathbf{r}^{l\ell_i}$, denoted by $\left\| {}^{l}\mathbf{r}^{l\ell_i} \right\|_2$. The bearing angle φ^{ℓ_i} corresponds to the angle between the vector ${}^{l}\mathbf{r}^{l\ell_i}$ and the first axis of $\mathcal{F}_L$. The angle is by convention positive when going counterclockwise from the first axis to the second axis of $\mathcal{F}_L$. The ranges and bearing angles can also be interpreted as the *polar coordinates* of the landmarks with respect to a *polar coordinate system* $\mathring{\mathcal{F}}_L$, whose *pole* is located at the point l and *polar axis* along the first axis of $\mathcal{F}_L$. Therefore, the measurement of a landmark ℓ_i can be represented by the column vector ${}^{\mathring{l}}\boldsymbol{\ell}_i = (r^{\ell_i}, \varphi^{\ell_i})^T$.

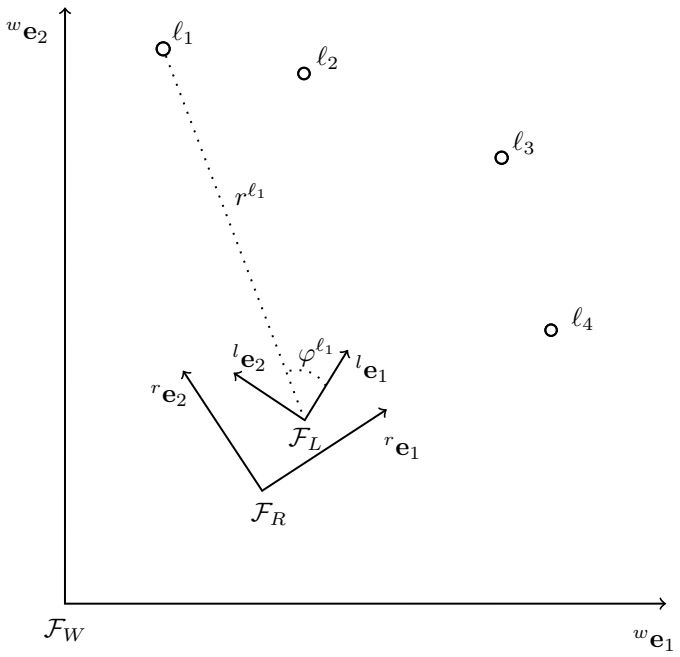


Figure 4.2: Setup of the coordinate frames for the differential-drive robot example. While moving with respect to $\mathcal{F}_W$, the robot observes landmarks in the environment with a laser rangefinder anchored at $\mathcal{F}_L$.

Considering the discussions in Sec. 3.4, we need to describe the function that maps the latent variables of the system to the observations. In our setup, the latent variables are ${}^w\mathbf{T}_r(t)$, ${}^r\mathbf{T}_l$, and ${}^w\boldsymbol{\ell} = \{{}^w\boldsymbol{\ell}_1, \ldots, {}^w\boldsymbol{\ell}_{Nl}\}$. At a given time T, we can express the coordinates of the landmark ℓ_i as

$$ {}^l\boldsymbol{\ell}_i(T) \quad = \quad {}^r\mathbf{T}_l^{-1}\ {}^w\mathbf{T}_r^{-1}(T)\ {}^w\boldsymbol{\ell}_i, \tag{4.11} $$

which can be simplified to

$$ {}^l\boldsymbol{\ell}_i(T) \quad = \quad {}^l\mathbf{T}_r\ {}^r\mathbf{T}_w(T)\ {}^w\boldsymbol{\ell}_i. \tag{4.12} $$

If we denote the left-hand side of Equ. 4.12 as ${}^l\boldsymbol{\ell}_i(T) = ({}^l\ell_{i_1}(T),\ {}^l\ell_{i_2}(T))^T$, we can formulate the measurement of the LRF as

$$ {}^{\dot{l}}\boldsymbol{\ell}_i(T) \quad = \quad \begin{pmatrix} r^{\ell_i}(T) \\ \varphi^{\ell_i}(T) \end{pmatrix} \quad = \quad {}^\circ f_\perp({}^l\boldsymbol{\ell}_i(T)), \tag{4.13} $$

where

$$ {}^\circ f_\perp({}^l\boldsymbol{\ell}_i(T)) \quad = \quad \begin{pmatrix} \sqrt{{}^l\ell_{i_1}(T)^2 \ + \ {}^l\ell_{i_2}(T)^2} \\ \text{atan2}({}^l\ell_{i_2}(T),\ {}^l\ell_{i_1}(T)) \end{pmatrix}. \tag{4.14} $$

The function ${}^{\circ}f_{\perp}(\cdot)$ converts from Cartesian to polar coordinates. Obviously, we can express a separate forward model $h_{\ell_i}(\cdot)$ for each landmark, that we define as

$$h_{\ell_i}({}^{w}\mathbf{T}_r(T),\ {}^{w}\boldsymbol{\ell}_i,\ {}^{r}\mathbf{T}_l) \quad = \quad {}^{\circ}f_{\perp}({}^{l}\mathbf{T}_r\ {}^{r}\mathbf{T}_w(T)\ {}^{w}\boldsymbol{\ell}_i). \tag{4.15}$$

The function $h_{\ell_i}(\cdot)$ can be interpreted as the composition of three functions, the inner part transforms the landmark coordinates from $\mathcal{F}_W$ to $\mathcal{F}_R$, the intermediary part transforms from $\mathcal{F}_R$ to $\mathcal{F}_L$, and the outer part transforms from $\mathcal{F}_L$ to $\mathcal{F}_L^{\circ}$.

Motion Model

As suggested in Sec. 4.1, the motion of the robot can be described by the following differential equations

$$\begin{aligned} {}^{w}\dot{\mathbf{r}}^{wr}(t) &= {}^{w}\mathbf{v}^{wr}(t), \\ \dot{\varphi}(t)\ {}^{w}\mathbf{u}(t) &= {}^{w}\boldsymbol{\omega}^{wr}(t), \end{aligned} \tag{4.16}$$

where ${}^{w}\mathbf{u}(t)$ is the instantaneous axis of rotation around which the robot rotates at rate $\dot{\varphi}(t)$. The motion being constrained to a plane, ${}^{w}\mathbf{u}(t)$ is constant and points in a direction perpendicular to the plane. Along the same lines, the angular velocity ${}^{w}\boldsymbol{\omega}^{wr}(t)$ is a scalar that we denote $\omega(t)$. We shall further express the linear velocity in $\mathcal{F}_R$ to give

$$\begin{aligned} {}^{w}\dot{\mathbf{r}}^{wr}(t) &= {}^{w}\mathbf{R}_r(t)\ {}^{r}\mathbf{v}^{wr}(t), \\ \dot{\varphi}(t) &= \omega(t), \end{aligned} \tag{4.17}$$

where

$${}^{w}\mathbf{R}_r(t) \quad = \quad \begin{pmatrix} \cos\varphi(t) & -\sin\varphi(t) \\ \sin\varphi(t) & \cos\varphi(t) \end{pmatrix}. \tag{4.18}$$

Due to the differential drive, the robot moves without lateral slipping. Therefore, we complete the motion model with the following *non-holonomic constraint*

$$\begin{pmatrix} -\sin\varphi(t) & \cos\varphi(t) \end{pmatrix}\ {}^{w}\dot{\mathbf{r}}^{wr}(t) \quad = \quad 0. \tag{4.19}$$

For the purpose of this application, we choose to approximate Equ. 4.17 in discrete time using a first-order *Runge-Kutta method* with small step sizes T as

$$\begin{aligned} {}^{w}\mathbf{r}^{wr}_{k+1} &= {}^{w}\mathbf{r}^{wr}_{k} + T\ {}^{w}\mathbf{R}_{r_k}\ {}^{r}\mathbf{v}^{wr}_{k}, \\ \varphi_{k+1} &= \varphi_k + T\omega_k, \\ 0 &= \begin{pmatrix} 0 & 1 \end{pmatrix}\ {}^{r}\mathbf{v}^{wr}_{k}, \end{aligned} \tag{4.20}$$

where k is the discrete-time index and

$$ {}^{w}\mathbf{R}_{r_k} = \begin{pmatrix} \cos\varphi_k & -\sin\varphi_k \\ \sin\varphi_k & \cos\varphi_k \end{pmatrix}. \tag{4.21} $$

Although the discrete-time model in Equ. 4.20 could be improved by a higher-order Runge-Kutta method, we consider it as an adequate approximation for small step sizes. We shall show in Sec. 4.3 an alternative formulation that exploits the continuous-time motion model.

The robot can measure its linear and angular velocity through *rotary encoders* attached to its wheels. We adopt here a simplified odometry model that provides at time k the linear velocity v_k^o and the angular velocity ω_k^o in an odometry frame $\mathcal{F}_O$ collocated with $\mathcal{F}_R$. Using Equ. 4.20, this suggests the following forward model for odometry measurements

$$ \begin{aligned} & h_o({}^{w}\mathbf{T}_{r_{k+1}}, {}^{w}\mathbf{T}_{r_k}, T) \\ & = \begin{pmatrix} T^{-1} \begin{pmatrix} 1 & 0 \end{pmatrix} {}^{r}\mathbf{R}_{w_k} \left({}^{w}\mathbf{r}_{k+1}^{wr} - {}^{w}\mathbf{r}_{k}^{wr} \right) \\ 0 \\ T^{-1}(\varphi_{k+1} - \varphi_k) \end{pmatrix}, \end{aligned} \tag{4.22} $$

where we have added a pseudo-constant output to satisfy the non-holonomic constraint. Here again, we notice that the first component of $h_o(\cdot)$ consists in the composition of functions.

Estimation Model

The forward models for the LRF and the odometry sensors being defined, we can now formulate the underlying probabilistic model. For this first application, we assume that we obtain measurements from each sensor synchronously, *i.e.*, for each timestep k, we observe $\mathbf{v}_k^o = (v_k^o, 0, \omega_k^o)^T$ and $\boldsymbol{\ell}_k = \{\boldsymbol{\ell}_{1_k}, \ldots, \boldsymbol{\ell}_{Nl_k}\}$, with $\boldsymbol{\ell}_{i_k} = (r_k^{\ell_i}, \varphi_k^{\ell_i})^T$. We shall examine how to withdraw this assumption in Sec. 4.3.

For each measurement $\mathbf{v}_k^o$, we can define an error term

$$ \mathbf{e}_k^o = \mathbf{v}_k^o - h_o({}^{w}\mathbf{T}_{r_{k+1}}, {}^{w}\mathbf{T}_{r_k}, T), \tag{4.23} $$

distributed as

$$ \mathbf{e}_k^o \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}^o), \tag{4.24} $$

with covariance matrix $\boldsymbol{\Sigma}^o = \text{diag}\left[\sigma_v^2, \sigma_c^2, \sigma_\omega^2\right]$. We add here a small variance σ_c^2 for the non-holonomic constraint. The attentive reader might have noticed that we have violated the objective Bayesian perspective pursued thus far. The lateral velocity constraints indeed act as a prior for ${}^{w}\mathbf{T}_{r_{1:N}}$. However, these latent variables are part of the nuisance parameters and the prior reflects a real physical phenomenon.

Similarly, we can express an error term for each $\boldsymbol{\ell}_{i_k}$ as

$$ \mathbf{e}_k^{\ell_i} = \boldsymbol{\ell}_{i_k} - h_{\ell_i}({}^{w}\mathbf{T}_{r_k}, {}^{w}\boldsymbol{\ell}_i, {}^{r}\mathbf{T}_l), \tag{4.25} $$

distributed as

$$\mathbf{e}_k^{\ell_i} \quad \sim \quad \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}^{\ell}), \tag{4.26}$$

with covariance matrix $\mathbf{\Sigma}^{\ell} = \text{diag}\left[\sigma_r^2, \sigma_\varphi^2\right]$.

With regard to the probability theory presented in Sec. 2.1.1, we henceforth adopt a simplified notation in order not to unnecessarily distract the reader, *i.e.*, we use the same notation for random variables or variates. Likewise, the distinction between deterministic and random quantities shall become clear from the context. A data sample consists in $\{\mathbf{v}_{1:N}^o, \boldsymbol{\ell}_{1:N}\}$, the calibration variable in $\mathbf{\Theta} = \{{}^r\mathbf{T}_l\}$, and the nuisance variable in $\mathbf{\Psi} = \{{}^w\mathbf{T}_{r_{1:N}}, {}^w\boldsymbol{\ell}\}$. The joint distribution over the random variables can be represented by the Bayesian network in Fig. 4.3. The estimation problem can thus be formulated as the computation of the marginal posterior density $f({}^r\mathbf{T}_l \mid \mathbf{v}_{1:N}^o, \boldsymbol{\ell}_{1:N})$, from the joint posterior density $f({}^r\mathbf{T}_l, {}^w\mathbf{T}_{r_{1:N}}, {}^w\boldsymbol{\ell} \mid \mathbf{v}_{1:N}^o, \boldsymbol{\ell}_{1:N})$.

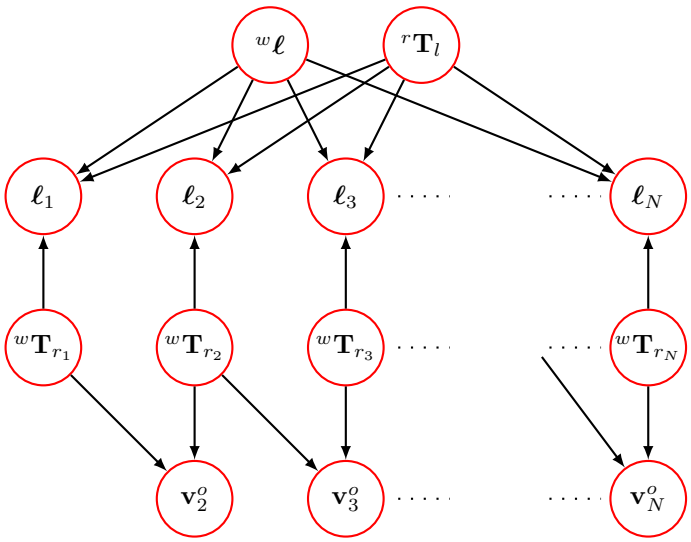


Figure 4.3: The Bayesian network representation of the factorization for the joint probability density function over data samples, calibration, and nuisance variables, for the laser rangefinder application. The observations and positions of the landmarks have been grouped into single variables for the purpose of the illustration. For each $\boldsymbol{\ell}_i$, the full model would require Nl additional arrows. Whenever a landmark is not visible, the corresponding arrow in the graphical model can be dropped.

We can obtain an initial estimate for ${}^w\mathbf{T}_{r_{2:N}}$ by successively applying the discretized motion model in Equ. 4.20 with the odometry measurements $\mathbf{v}_{1:N}^o$ starting from an arbitrary location ${}^w\mathbf{T}_{r_1}$. This process if referred to as *odometry integration*. The initial guess for ${}^r\mathbf{T}_l$ comes from a measuring tape and an angle finder. For ${}^w\boldsymbol{\ell}$, we can use the inverse model of Equ. 4.15 while setting the initial guesses for ${}^w\mathbf{T}_{r_{1:N}}$ and ${}^r\mathbf{T}_l$.

Jacobian Matrices

To complete our setup, we require the analytical Jacobian matrices for $\mathbf{e}_k^o$ and $\mathbf{e}_k^{\ell_i}$. This task is grandly facilitated by the form of $h_o(\cdot)$ and $h_{\ell_i}(\cdot)$. Indeed, both functions can be expressed in terms of compositions of functions, *i.e.*, we can use the *chain rule for derivatives*, which states that the total Jacobian becomes the product of the Jacobian matrices of the composed functions.

In this application, we parameterize an affine transformation with the tuple (r_1, r_2, φ), such that a transformation matrix can be written as

$$\mathbf{T} = \begin{pmatrix} \cos\varphi & -\sin\varphi & r_1 \\ \sin\varphi & \cos\varphi & r_2 \\ 0 & 0 & 1 \end{pmatrix}. \tag{4.27}$$

In our setup, the inverse affine transformation is applied to a point $\mathbf{p} = (p_1, p_2, 1)^T$ as $\mathbf{T}^{-1}\mathbf{p}$. Consequently, we can compute the Jacobian of this transformation with respect to $\mathbf{p}$ as

$$\mathbf{J}_{\mathbf{p}}^{\mathbf{T}^{-1}}(\varphi) = \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix}, \tag{4.28}$$

with respect to $\mathbf{r} = (r_1, r_2)^T$ as

$$\mathbf{J}_{\mathbf{r}}^{\mathbf{T}^{-1}}(\varphi) = \begin{pmatrix} -\cos\varphi & -\sin\varphi \\ \sin\varphi & -\cos\varphi \end{pmatrix}, \tag{4.29}$$

and with respect to φ as

$$\mathbf{J}_{\varphi}^{\mathbf{T}^{-1}}(\mathbf{p}, \mathbf{r}, \varphi) = \begin{pmatrix} \sin\varphi(-p_1 + r_1) + \cos\varphi(p_2 - r_2) \\ \cos\varphi(-p_1 + r_1) + \sin\varphi(-p_2 + r_2) \end{pmatrix}. \tag{4.30}$$

For the sake of clarity, we have omitted the rows and columns related to the homogeneous component.

For the Cartesian to polar coordinates function ${}^{\circ}f_{\perp}(\cdot)$, the Jacobian matrix can be expressed as

$$\mathbf{J}_{\mathbf{p}}^{{}^{\circ}f_{\perp}}(\mathbf{p}) = \begin{pmatrix} p_1(p_1^2 + p_2^2)^{-\frac{1}{2}} & p_2(p_1^2 + p_2^2)^{-\frac{1}{2}} \\ -p_2(p_1^2 + p_2^2)^{-1} & p_1(p_1^2 + p_2^2)^{-1} \end{pmatrix}. \tag{4.31}$$

From these expressions, we can derive the entire set of Jacobian matrices required for our estimation problem.

4.2.2 Observability Analysis

Before we delve into experimental evaluation, we shall perform an a priori analysis of the model. After an intuitive observability analysis, we shall validate our thoughts by the examination of the Jacobian matrix structure and the cost function graph. We note that this analysis is not necessary for our algorithm to work properly, but nevertheless provides valuable insights for interpreting the experimental results and for a proper understanding of the behavior of our method.

Joint Forward Model

We shall first start this analysis by considering a subset of the estimation problem, *i.e.*, the estimation of robot's poses using odometry measurements. This simple example aims at illustrating the procedure that will drive the observability analysis of the full estimation problem.

Recalling the discussions in Sec. 2.1.7, observability pertains to the injectivity of the forward model. Assuming a sample of odometry data $\mathbf{v}^o_{1:N}$, we inspect the joint forward model $\mathbf{v}^o_{1:N} = H_o({}^w\mathbf{T}_{r_{1:N}}, T)$ that generates it. In $H_o(\cdot)$, the step size T is constant and ${}^w\mathbf{T}_{r_{1:N}}$ form the domain of the function. If distinct instances of ${}^w\mathbf{T}_{r_{1:N}}$ map to the same outputs $\mathbf{v}^o_{1:N}$, the parameters (*states* in robotic or control terms) are not observable. In other words, the inverse function $H_o^{-1}(\cdot)$ is not uniquely determined by $H_o(\cdot)$. In that case, if the function $H_o(\cdot)$ represents the expected value of a data distribution, such as in the regression models of Sec. 2.1.6, the associated statistical model is not identifiable.

For the sake of the example, if we withdraw the non-holonomic constraint on the lateral linear velocity of the robot, there is an infinite number of robot's poses ${}^w\mathbf{T}_{r_{1:N}}$ that generate the same odometry outputs. For instance, the set of robot's poses with identical orientations along the second axis of the robot's frame will always be reflected by a constant zero output on the odometry sensors. Consequently, robot's poses are not observable. On the other hand, the zero velocity constraint on the second component of the linear velocity in the robot frame disallows these configurations and results in full observability. Our informal explanation is in line with the analysis conducted in [MSS06], which suggests a pose parameterization in polar coordinates to solve the problem.

Along this line of thoughts, we can proceed to the full joint forward model including the landmark positions and calibration parameters, *i.e.*,

$$\begin{pmatrix} \mathbf{v}^o_{1:N} \\ \boldsymbol{\ell}_{1:N} \end{pmatrix} = H({}^w\mathbf{T}_{r_{1:N}}, T, {}^w\boldsymbol{\ell}, {}^r\mathbf{T}_l). \tag{4.32}$$

The model being under-constrained, there is an infinite number of elements of the domain of $H(\cdot)$ that map to the same element in its codomain. For instance, any affine transformation applied to ${}^w\mathbf{T}_{r_{1:N}}$, ${}^w\boldsymbol{\ell}$, and ${}^r\mathbf{T}_l$, gives rise to identical outputs. If the relative poses ${}^w\mathbf{T}_{r_{1:N}}$ have the same orientation, an arbitrary translation applied to ${}^r\mathbf{T}_l$ and to ${}^w\boldsymbol{\ell}$ with respect to $\mathcal{F}_W$ has no influence on the outputs. This situation corresponds to the robot driving in a straight line or standing still, with the LRF coordinate system translated along with the landmark positions. Here also, we notice that the linear velocity constraint proscribes some unobservable configurations. Although this rough analysis does not assert an exact figure, the identification of these unobservable directions can guide our experimental evaluation.

In the physical world, the inputs and outputs of $H(\cdot)$ are random quantities associated with a continuous probability density function. As we pointed out in Sec. 2.1.1, the probability for the realization of any of these configurations is zero by definition. However, as we might be arbitrarily close to one of them, our estimation problem will appear to be ill-conditioned. For this reason, it only makes sense to speak about numerical observability when a real physical system is involved. We might even extend this intuition to an apparently noise-free simulation of a physical system on a computer. If we evaluate the

forward model $H(\cdot)$ repeatedly by varying the parameters in the unobservable direction, the outputs will most likely be normally distributed with a standard deviation proportional to the unit roundoff. Even though the numerical noise is not random, the accumulation of its small effects transforms a supposedly deterministic output into a normally distributed random variate. It can be seen as a good example of the occurrence of the central limit theorem in practice. We shall investigate these aspects in more details when we come to the simulated experiments in Sec. 4.2.3.

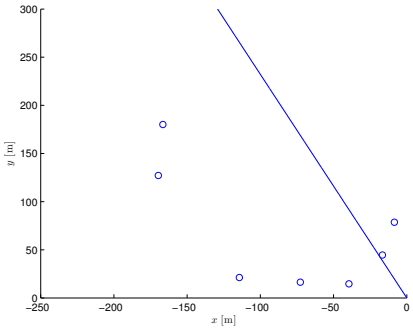
Jacobian Matrix Structure

Following on this informal analysis, we shall now examine if the structure of the Jacobian matrix supports our claims regarding the identified unobservable configurations. As can be understood from the model in Sec. 4.2.1, no data appears in the Jacobian matrix. We can thus evaluate it for any realization of the latent variables given an unobserved data sample.

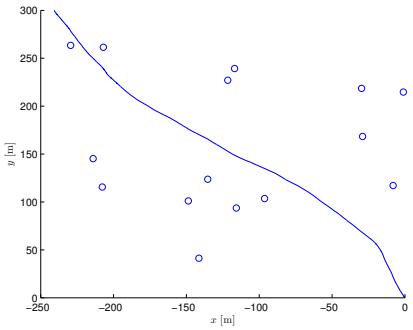
In a first effort, we evaluate the full Jacobian matrix $[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]$ for the configurations depicted in Fig. 4.4. The landmark positions ${}^{w}\boldsymbol{\ell}$ are sampled uniformly within a region around the robot's trajectory ${}^{w}\mathbf{T}_{r_{1:N}}$ and the calibration variable ${}^{r}\mathbf{T}_{l}$ is chosen arbitrarily. While the trajectory is perfectly straight in Fig. 4.4a, we have changed the setup in Fig. 4.4b by applying a slight deviation on ${}^{w}\mathbf{T}_{r_{1:N}}$. Given that our initial guess for ${}^{w}\mathbf{T}_{r_{1:N}}$ is based on integrated odometry, this is a typical situation we shall encounter when starting the iterative optimization algorithm. From a linear algebra perspective, this situation illustrates the perturbation theory presented in Sec. 2.2.2, *i.e.*, the evaluated Jacobian matrix is a perturbed version of the previous one. We shall finally focus on the situation sketched in Fig. 4.4c. The robot's trajectory ${}^{w}\mathbf{T}_{r_{1:N}}$ is modeled as a sinusoidal path and hence contains multiple changes of orientation.

By selecting a reasonable data sample size, we can analyze the structure of the full Jacobian matrix by SVD. The computed singular values for these three configurations are shown in Fig. 4.5a. For the three cases, the three singular values clamped at zero correspond to the degrees of freedom (two for translation and one for orientation) induced by the lack of global anchor of the model. In the straight trajectory case, two singular values associated with the translation of ${}^{r}\mathbf{T}_{l}$ and ${}^{w}\boldsymbol{\ell}$ remain at zero. In the perturbed straight trajectory, these two singular values depart from zero, which shall lead to an incorrect rank determination. The sinusoidal path being well-behaved, the singular values are legitimately above zero. Here, we notice the difficulty in assessing the right threshold for the numerical rank. As a comparison, we display in Fig. 4.5b the singular values obtained by scaling the columns of the Jacobian matrix.

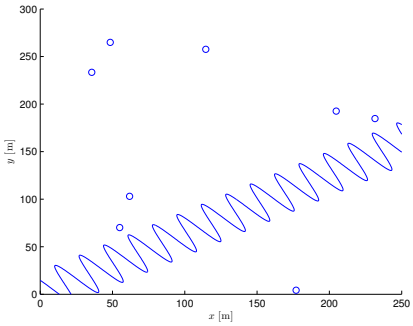
After the SVD analysis of the full Jacobian matrix $[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]$, we display in Fig. 4.6 the singular values of the reduced matrix $\mathbf{A}_{\boldsymbol{\theta}}$ from Equ. 3.17. The plot in Fig. 4.6b with the scaled singular values is particularly encouraging, since it demonstrates the feasibility of numerical rank detection using the same threshold for the three trajectories (around 0.02 in this case), as compared to Fig. 4.5b where no such threshold exists. We shall elaborate further on that topic in Sec. 4.2.3 and analyze the effect of noise on the singular values.



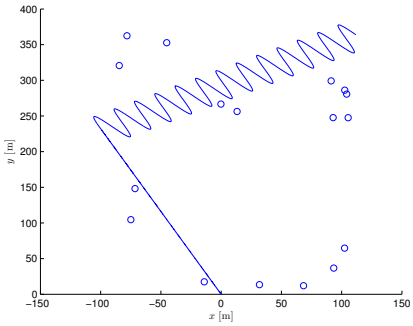
(a) Straight trajectory.



(b) Perturbed straight trajectory.

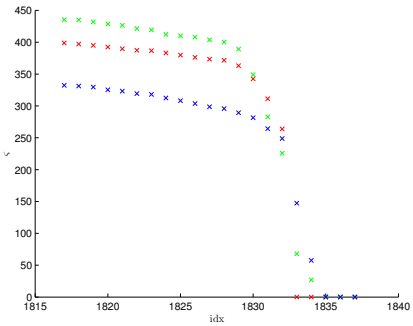


(c) Sinusoidal trajectory.

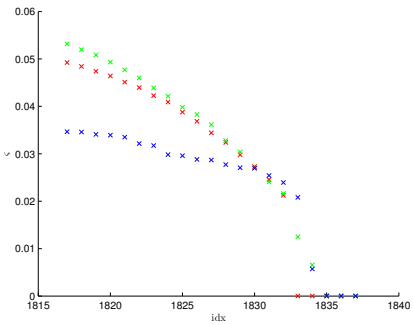


(d) Straight-sinusoidal trajectory.

Figure 4.4: Examples of robot's trajectories. The blue line corresponds to the robot's path ${}^{w}\mathbf{T}_{r_{1:N}}$ and the blue circles to the landmark positions ${}^{w}\boldsymbol{\ell}$. In Fig. 4.4a, only the rotational part of ${}^{r}\mathbf{T}_{l}$ is observable. In Fig. 4.4b, the translational part of ${}^{r}\mathbf{T}_{l}$ appears observable due to noise in the Jacobian matrix. Fig. 4.4c illustrates the fully observable case and Fig. 4.4d the combination of a partially and fully observable configuration.

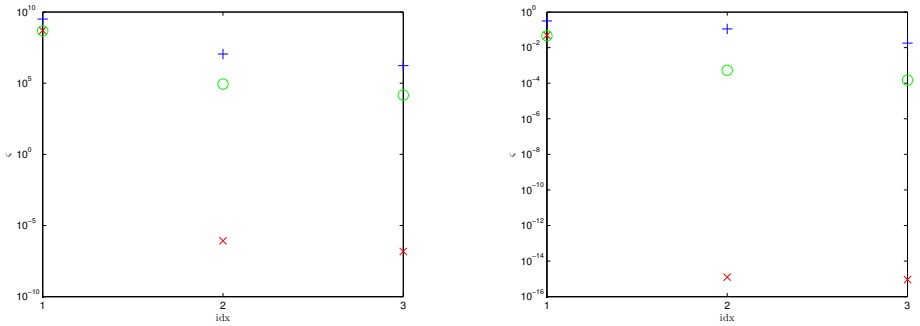


(a) Singular values.



(b) Scaled singular values.

Figure 4.5: The 20 smallest singular values of $[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]$ for the straight (red), perturbed straight (green), and sinusoidal (blue) trajectory. Fig. 4.5a shows the singular values of the original Jacobian matrix and Fig. 4.5b of the scaled Jacobian matrix using Equ. 3.58.



(a) Singular values. (b) Scaled singular values.

Figure 4.6: Singular values of $\mathbf{A}_{\boldsymbol{\theta}}$ for the straight (red), perturbed straight (green), and sinusoidal (blue) trajectory. Fig. 4.6a shows the singular values of the original $\mathbf{A}_{\boldsymbol{\theta}}$ and Fig. 4.6b of the scaled $\mathbf{A}_{\boldsymbol{\theta}}$ using Equ. 3.58.

Cost Function Graph

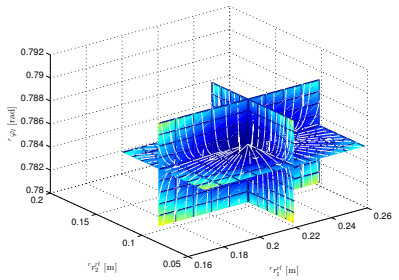
To conclude this a priori analysis, it is worth picturing the graph of the cost function as a function of the calibration parameters $\boldsymbol{\theta}$, especially under the unobservable conditions. This graph shall reflect the difficulty in locating a local minimum for $\boldsymbol{\theta}$.

In Fig. 4.7a, we have fixed the nuisance variables ${}^{w}\mathbf{T}_{r_{1:N}}$ and ${}^{w}\boldsymbol{\ell}$ corresponding to the sinusoidal path in Fig. 4.4c and evaluated the cost function at different values of the calibration variable ${}^{r}\mathbf{T}_{l}$. In this fully observable situation, we notice that the cost function is well-behaved, *i.e.*, the gradient lines are convergent. To illustrate the behavior of the cost function in a partially observable setting, we display in Fig. 4.7b the cost function corresponding to the straight trajectory in Fig. 4.4a. In this plot, we have applied the same translation to ${}^{r}\mathbf{T}_{l}$ and ${}^{w}\boldsymbol{\ell}$, resulting in identical costs for the translation parameters of ${}^{r}\mathbf{T}_{l}$. Consequently, the gradient lines in this plane are divergent and a nonlinear optimization algorithm without regularization will most likely fail. Finally, Fig. 4.7c demonstrates the numerically unobservable situation depicted in Fig. 4.4b. Although the input data contains just a slight deviation of the previous case, we witness here that the cost function appears well-behaved.

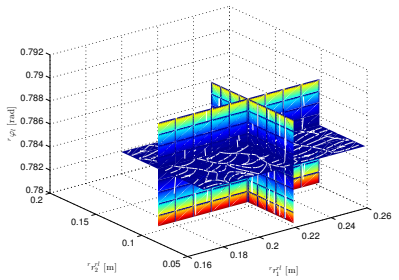
4.2.3 Simulated Experiments

For this first application of our algorithm, we shall carry out extensive simulated experiments to confirm the different claims we have formulated throughout the thesis. Simulated results have to be interpreted cautiously though, as they will only convey the validity or inconsistency of our approach up to the approximate model representing the underlying real-world process.

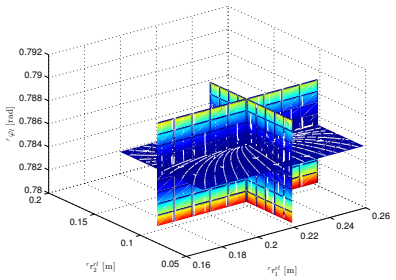
As we have an analytical formulation of the forward model for each sensor, we can select any configuration of the latent variables and generate measurements using their data distribution. By modifying the variances, we are also able to analyze the robustness of our algorithm to various noise levels. Obviously, we have to generate sets of poses ${}^{w}\mathbf{T}_{r_{1:N}}$ compatible with the robot's physical capabilities. We have chosen here a sinusoid function, parameterized by an amplitude A and a frequency f, providing at will straight to highly curved



(a) Well-behaved cost function.



(b) Ill-behaved cost function.



(c) Numerically ill-behaved cost function.

Figure 4.7: Cost function for the fully observable (Fig. 4.7a), partially observable (Fig. 4.7b), and numerically unobservable (Fig. 4.7c) scenario. The planes are located at the ground truth value for the calibration parameters and the gradient of the cost function is represented by the white arrows.

trajectories. The sampled odometry measurements are integrated to give the initial guesses ${}^{w}\mathbf{T}_{r_{1:N}}^{(0)}$ for the robot's trajectories. The landmark positions ${}^{w}\boldsymbol{\ell}$ get uniformly sampled within a reasonable support region determined by the starting and ending points of the robot's trajectories. To create an initial guess ${}^{r}\mathbf{T}_{l}^{(0)}$ for the calibration variable, we either take a normal sample of ${}^{r}\mathbf{T}_{l}$ or assign a fixed value around it. This latter case is more appropriate when it comes to the comparison of our algorithm with other methods, as we require similar initial conditions. Using ${}^{w}\mathbf{T}_{r_{1:N}}^{(0)}$ and ${}^{r}\mathbf{T}_{l}^{(0)}$, we can finally produce an initial guess ${}^{w}\boldsymbol{\ell}^{(0)}$ for ${}^{w}\boldsymbol{\ell}$ with its inverse model. We average here the computed landmark positions over a couple of different poses, preferably at the start of the trajectory.

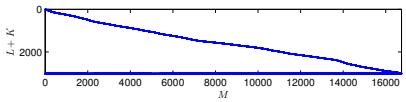
For the rest of this section, we shall commence with a practical complexity analysis of our algorithm. In a second stage, we conduct a simulation targeted at validating the normal assumption for the posterior distribution. We will then investigate the evolution over time of the information gain, which is a key measure in the online part of our algorithm. We shall then study the robustness of the numerical rank detection to varying levels of noise. Lastly, we conclude with some qualitative results and comparisons with other methods.

Complexity Analysis

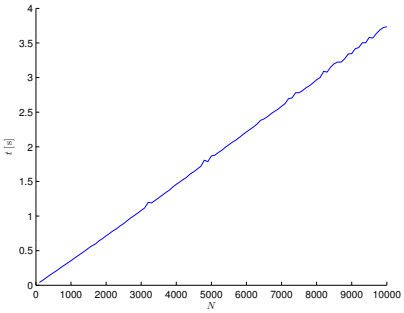
In Sec. 3.4, we have formulated some theoretical figures regarding the time ($\mathcal{O}(L^{\frac{3}{2}})$) and storage ($\mathcal{O}(L)$) complexity of our algorithm. The storage complex-

ity is validated by design in this application, since we add a constant number of nonzero elements to the Jacobian matrix for each measurement. As an example, we show in Fig. 4.8a the sparsity pattern of $[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]^T$ for 100 seconds of robot's driving.

Regarding time complexity, we have evaluated the computation time of Alg. 3.2 for one iteration of the nonlinear optimization, averaged over 10 repetitions, for a growing number of samples. In Fig. 4.8b, we witness that the time complexity is indeed close to linear time. For this experiment, we have used a step size of $T = 0.1$ [s], *i.e.*, the largest sample size N corresponds to a robot's trajectory of approximatively 17 minutes. Furthermore, we have employed a single-threaded implementation for the Jacobian evaluations. As will be demonstrated later in this section, the input size to Alg. 3.2 will usually be way below the largest N displayed in the plot, due to our information selection scheme.



(a) Sparsity pattern.



(b) Computation time.

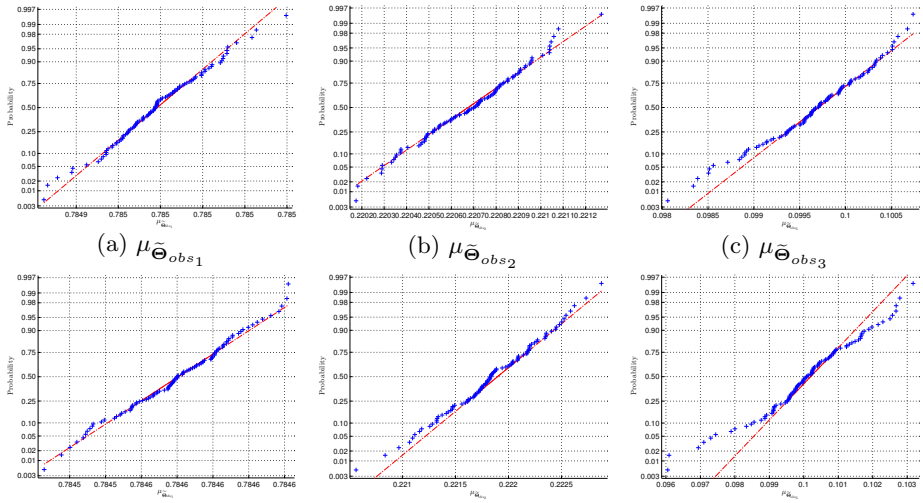
Figure 4.8: Sparsity pattern of $[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]^T$ for 100 seconds of robot's driving (Fig. 4.8a) and evolution of the computation time as a function of the data sample size (Fig. 4.8b).

Normal Approximation

In Sec. 3.1, we claimed that the posterior distribution over the nuisance and calibration variables, along with the posterior over the observable part of the calibration variable, can be approximated through Laplace's method by a normal distribution. As this assumption is essential to the computation of the information gain, it deserves verification. We will inspect the shape of these posteriors for small and large batch sizes, and for full and partial observability of the calibration parameter.

There exists various tools for evaluating the normality of a distribution, either graphical or statistical. The most intuitive method is to visualize an *histogram plot* or a *scatter plot* of the samples. However, a *normal probability plot* provides more information. The normal probability plot is a *Q-Q plot*, *i.e.*, it relates the theoretical quantiles of the normal distribution to the sample quantiles. If the sample distribution is normal, the plot should show approximatively a straight line. For multivariate data, as in our case, we can either display each dimension separately or visualize a Q-Q plot against a chi-squared distribution, by taking the squared Mahalanobis distances of the samples using their sample covariance.

Using the setup depicted in Fig. 4.4c, we simulate data samples of size $N = 1000$ and $N = 200$ that we assemble into batches $\mathcal{D} = \{\mathbf{v}^o_{1:N}, \boldsymbol{\ell}_{1:N}\}$ and request estimates $\widehat{\boldsymbol{\theta}}$ to Alg. 3.2. While collecting the estimates, we repeat the experiment 100 times and construct the normal probability plots in Fig. 4.9 for each dimension of the observable subspace. This experiment shall be representative for the full observability of the calibration parameters. From the plots, we observe that the normal approximation is satisfied even for smaller data sample sizes. It is worth noting that the observability of the parameters decreases from left to right in Fig. 4.9, hence explaining the slight deviation in the third column.



(a) $\mu_{\widetilde{\boldsymbol{\Theta}}_{obs_1}}$ (b) $\mu_{\widetilde{\boldsymbol{\Theta}}_{obs_2}}$ (c) $\mu_{\widetilde{\boldsymbol{\Theta}}_{obs_3}}$

(d) $\mu_{\widetilde{\boldsymbol{\Theta}}_{obs_1}}$ (e) $\mu_{\widetilde{\boldsymbol{\Theta}}_{obs_2}}$ (f) $\mu_{\widetilde{\boldsymbol{\Theta}}_{obs_3}}$

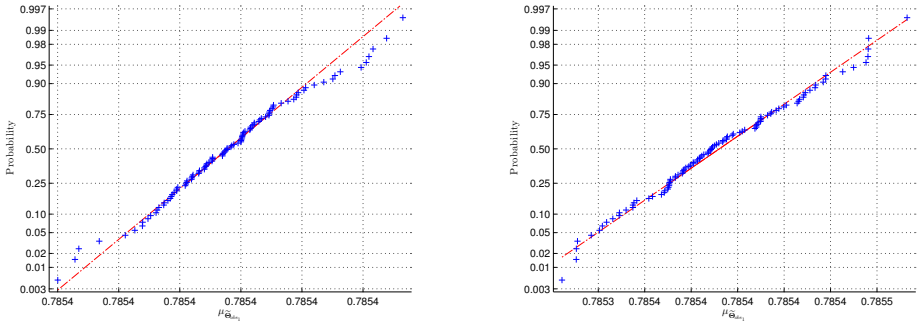
Figure 4.9: Normal probability plots for 100 samples (blue crosses) of $\widetilde{\boldsymbol{\Theta}}_{obs} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N$ with a data sample size of $N = 1000$ (top row) and $N = 200$ (bottom row). The observable subspace is of dimension 3.

Changing the experimental conditions to the setup in Fig. 4.4a, we perform the same experiment and display the normal probability plots in Fig. 4.10. This scenario illustrates the partial observability of the calibration parameters with an observable subspace of dimension one. Here also, the results are in line with our expectation.

Information Gain

Following on the previous section, the information gain computation relies on the normality assumption, referring to the method described in Sec. 3.3. In this section, we have also justified the pertinence of our online algorithm by the convergence of the information gain to a steady value as the sample size grows. We shall here evaluate the soundness of this argument by means of simulation.

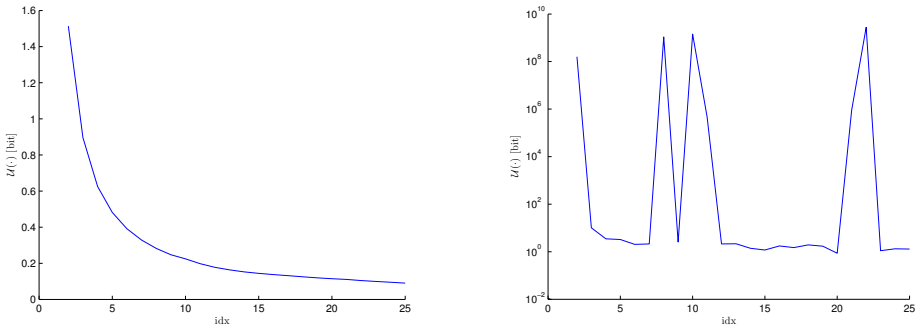
Starting with the well-behaved scenario of Fig. 4.4c, we collect data batches $\mathcal{D}_{1:N}$ of size $\Delta N = 200$ that we sequentially insert into Alg. 3.1. Instead of rejecting batches in Alg. 3.4, we gather the computed information gains, either with the entropy difference in Equ. 3.45 or the KL divergence in Equ. 3.46.



(a) $\mu_{\widetilde{\Theta}_{obs_1}}$ (b) $\mu_{\widetilde{\Theta}_{obs_1}}$

Figure 4.10: Normal probability plots for 100 samples (blue crosses) of $\widetilde{\Theta}_{obs} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N$ with a data sample size of $N = 1000$ (left) and $N = 200$ (right). The observable subspace is of dimension 1.

We repeat this experiment 10 times and report the averaged evolution of the information gains with respect to the incoming batches in Fig. 4.11. The plot in Fig. 4.11a shows that the entropy difference is an appropriate measure for the utility of an incoming batch and we shall henceforth adopt it in Alg. 3.4. Although promising theoretically, the instable behavior of the KL divergence can be explained by the small variances of the distributions which amplify insignificant changes in the means.

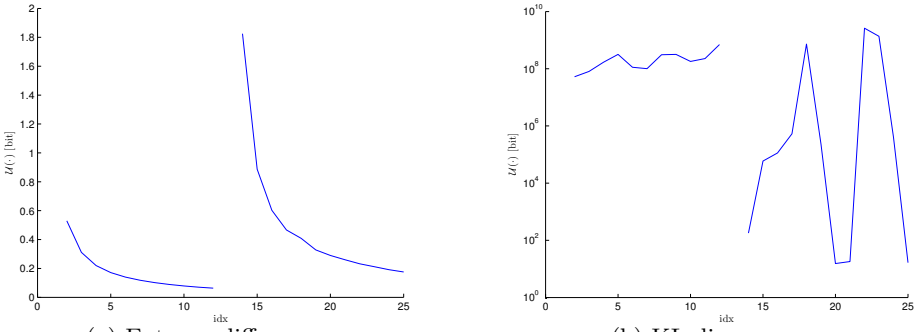


(a) Entropy difference. (b) KL divergence.

Figure 4.11: Evolution of the information gains with respect to incoming batches in a fully observable setting. While Fig. 4.11a uses the entropy difference to compute the utility $\mathcal{U}(\cdot)$ of the incoming batch, Fig. 4.11b employs the KL divergence.

In a second experiment, we aim at visualizing the information gain evolution with the setup depicted in Fig. 4.4d. In this scenario, the robot initially drives in a straight line and finishes with a sinusoidal trajectory. Consequently, this setup entails a combination of partial and full observability of the calibration parameters. Under these conditions, the information gains are plotted in Fig. 4.12. Here again, we notice the instability of the KL divergence in Fig. 4.12b. On the other hand, the information difference in Fig. 4.12a perfectly follows our intuition. It monotonically converges during the straight portion of the path

and increases when the remaining parameters become observable. As we set the information gain to infinity when the observable subspace dimension increases, the curve becomes discontinuous upon entering the sinusoidal portion of the path.



(a) Entropy difference. (b) KL divergence.

Figure 4.12: Evolution of the information gains with respect to incoming batches in a partially to fully observable setting. While Fig. 4.12a uses the entropy difference to compute the utility $\mathcal{U}(\cdot)$ of the incoming batch, Fig. 4.12b employs the KL divergence.

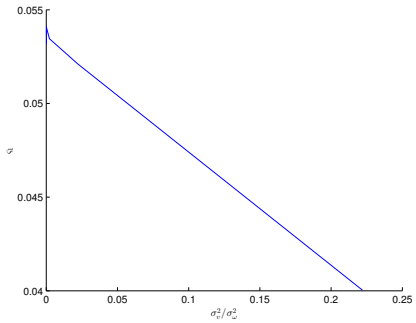
Numerical Rank and Observability Detection

A crucial statement in Chap. 3 concerns the ability of our algorithm to cope with locally unobservable parameters by analyzing the Jacobian matrix structures. Although this property has been indirectly demonstrated in the previous experiments, we shall examine here in more details the conditions under which it applies.

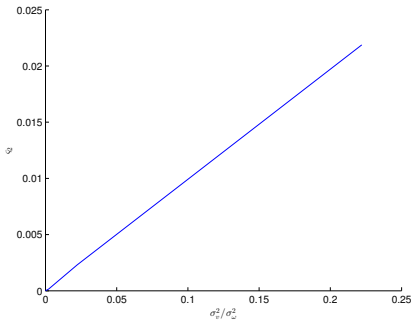
In a first experiment, we report in Fig. 4.13 the evolution of the singular values of the scaled $\mathbf{A}_{\boldsymbol{\theta}}$ as a function of the odometry noise in the straight trajectory example depicted in Fig. 4.4a. For each noise level, we have averaged 10 simulations runs of 500 seconds. The odometry variances σ_v^2 and σ_ω^2 range from 0 to 0.2. In the three plots, we observe that the singular values evolve linearly with the noise. For ς_1, the noise induces a slight decrease, which shall play no role in the numerical rank determination. Since we know from the analysis in Sec. 4.2.2 that ς_2 and ς_3 should be zero, an appropriate threshold for $r_{\epsilon_{\boldsymbol{\theta}}}$ will be in this case $\epsilon_{\boldsymbol{\theta}} \approx \sigma_v^2/10$.

Under identical experimental conditions, we show in Fig. 4.14 the evolution of the 2-norm and Frobenius norm of the perturbation matrix $\mathbf{E}$ defined in Sec. 2.2.2. We initially store the noise-free version of $\mathbf{A}_{\boldsymbol{\theta}}$ and compute $\mathbf{E}$ by subtracting it from the noisy matrices $\mathbf{A}^*_{\boldsymbol{\theta}}$. We also notice here that the odometry noise acts linearly on the perturbation matrix norms. Recalling the perturbation matrix theory which states that any singular value below $\|\mathbf{E}\|_2$ should be treated as zero, we can derive a similar threshold $\epsilon_{\boldsymbol{\theta}} \approx \sigma_v^2/10$ for the numerical rank. Therefore, we have demonstrated the validity of the numerical rank theory presented in Sec. 2.2.2 for the purpose of our application.

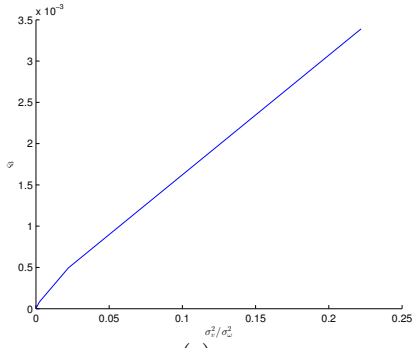
To conclude this part on the numerical rank and observability, we display in Fig. 4.15 the evolution of the singular values of the scaled $\mathbf{A}_{\boldsymbol{\theta}}$ as a function



(a) ς_1

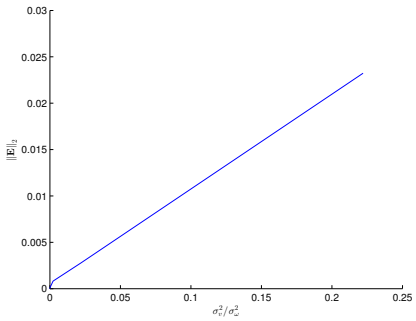


(b) ς_2

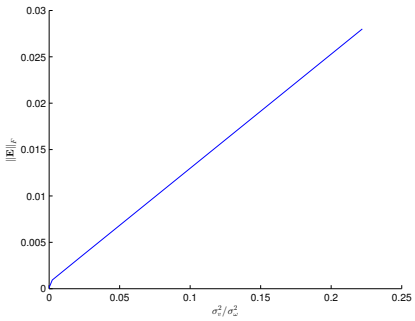


(c) ς_3

Figure 4.13: Evolution of the singular values $\varsigma_{1:3}$ of the scaled $\mathbf{A}_{\boldsymbol{\theta}}$ as a function of odometry noise σ_v^2 and σ_ω^2 on a straight trajectory.



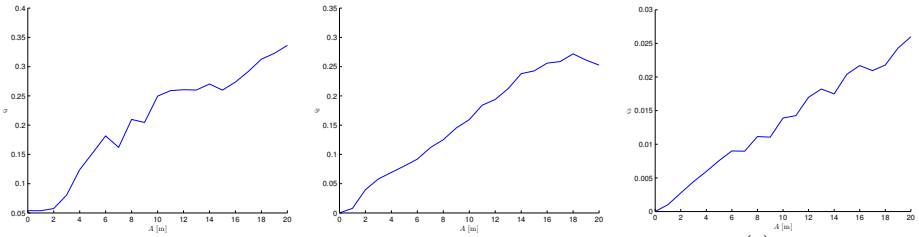
(a) 2-norm of $\mathbf{E}$.



(b) Frobenius norm of $\mathbf{E}$.

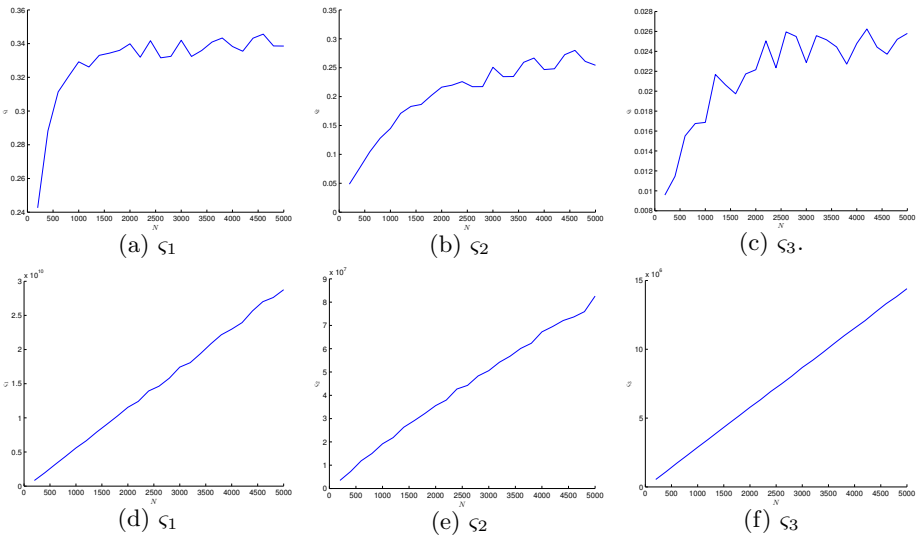
Figure 4.14: Evolution of the perturbation matrix 2-norm (Fig. 4.14a) and Frobenius norm (Fig. 4.14b) as a function of odometry noise σ_v^2 and σ_ω^2 on a straight trajectory.

of the amplitude A of the sinusoidal trajectory for a fixed sample size $N = 5000$, $\sigma_v^2 = 10^{-4}$, and $\sigma_\omega^2 = 10^{-4}$. According to the above discussions, the numerical rank threshold should be $\epsilon_{\boldsymbol{\theta}} \approx 10^{-5}$ for these variances. We observe that the calibration parameters become fully observable for an amplitude of approximatively $A = 0.5$ [m]. In Fig. 4.16, we have fixed the amplitude of the sinusoidal path to $A = 20$ [m] and varied the data sample size N. We see that the scaling of the Jacobian matrices dampens the effect of the data sample size on the singular values, which makes the numerical rank threshold $\epsilon_{\boldsymbol{\theta}}$ independent of N.



(a) ς_1 (b) ς_2 (c) ς_3

Figure 4.15: Evolution of the singular values $\varsigma_{1:3}$ of the scaled $\mathbf{A}_{\boldsymbol{\theta}}$ as a function of the amplitude A of a sinusoidal path.



(a) ς_1 (b) ς_2 (c) ς_3.

(d) ς_1 (e) ς_2 (f) ς_3

Figure 4.16: Evolution of the singular values $\varsigma_{1:3}$ of the scaled (top row) and unscaled (bottom row) $\mathbf{A}_{\boldsymbol{\theta}}$ as a function of the data sample size N on a sinusoidal path.

In all our experiments, the numerical rank $r_{\epsilon_{\boldsymbol{\psi}}}$ of $\mathbf{J}_{\boldsymbol{\psi}}$ (rank deficiency of 3 for the global transformation of all the frames) was correctly detected by the RRQR decomposition without a particular attention to the threshold $\epsilon_{\boldsymbol{\psi}}$ determined by Equ. 3.62 with the machine epsilon. Apparently, apart from the roundoff errors, sensor noise plays a minor role in the Jacobian matrix of the nuisance variables.

Qualitative Evaluation

After the detailed analysis of the different parts of our online self-calibration algorithm, we shall now demonstrate its overall behavior on some illustrative examples. It is for instance of interest to monitor the measurement batches considered as informative and whether the selection is consistent with our intuition. We shall also visualize the evolution of the posterior distribution over the calibration parameters as new batches are added.

We will stick to the three scenarios presented thus far, *i.e.*, a straight trajectory, a straight trajectory followed by a sinusoidal path, and a sinusoidal path. The algorithmic parameters used in the three experiments are summarized in Tab. 4.1. The sampled sensor data being normally distributed by design, we have omitted here the M-estimator parameters. The simulation parameters are reported in Tab. 4.2.

ΔN	δ	u_s	$u_{\boldsymbol{\psi}}$	$\epsilon_{\boldsymbol{\theta}}$	t_{max}	ε
200	0.2	10^{-16}	10^{-16}	10^{-5}	20	10^{-4}

Table 4.1: Algorithmic parameters for the experiments. We refer the reader to Sec. 3.4 for the signification of these parameters.

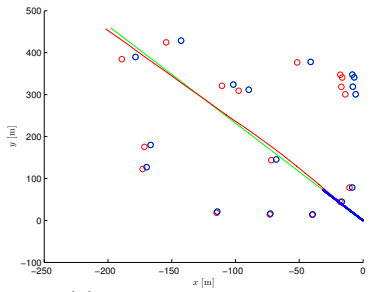
Parameter	Value
N	5000
T	0.1 [s]
$\sigma_v^2, \sigma_\omega^2, \sigma_r^2, \sigma_\varphi^2$	10^{-4}
σ_c^2	10^{-6}
Nl	17
${}^r\mathbf{T}_l$	${}^r r_1^{rl} = 0.219$ [m], ${}^r r_2^{rl} = 0.1$ [m], ${}^r\varphi_l = \pi/4$ [rad]

Table 4.2: Simulation parameters for the experiments. More details in Sec. 4.2.1.

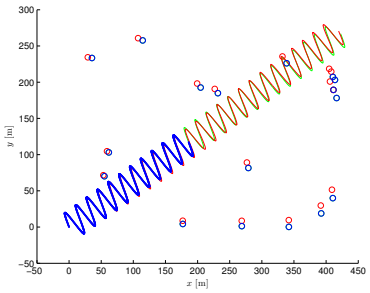
In Fig. 4.17, we show the selected informative batches, along with the estimated robot's poses and landmark positions. As mentioned in Sec. 3.1, these estimates are not of primary interest. Nevertheless, their proximity to the ground truth is a pleasing indicator of the applicability of our algorithm. For visualization, the lack of absolute anchor of the model has been compensated by an algorithm from [Fio01]. Out of 25 batches, the straight trajectory required 4, the sinusoidal trajectory 11, and the straight-sinusoidal trajectory 15.

In order to reveal the effect of the batch selection on the calibration parameters, we display in Fig. 4.18, the evolution of the parameter values as a function of the incoming batches. As expected, we witness that the parameters remain at their initial guess until they become observable.

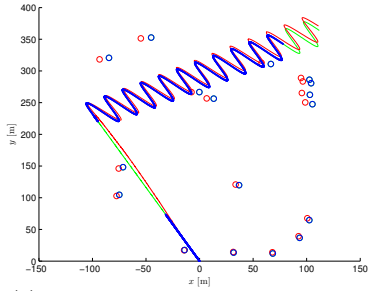
Finally, we show in Fig. 4.19 the evolution of the information gain as new batches are added to the online estimator. The behavior of the entropy difference concurs with the above experiments on the information gain.



(a) Straight trajectory.

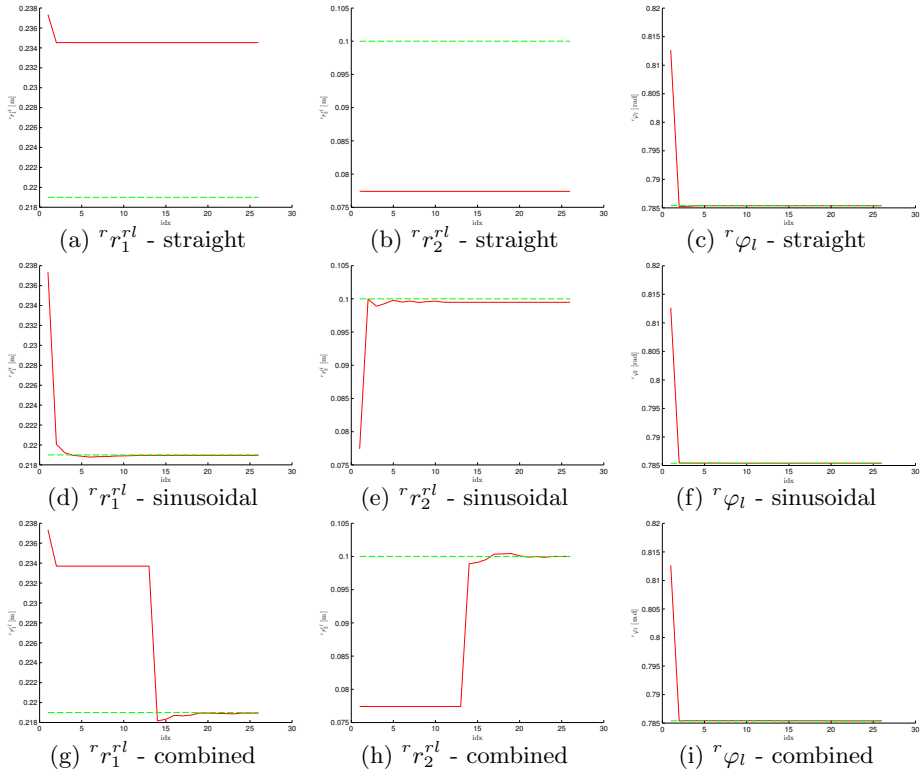


(b) Sinusoidal trajectory.



(c) Straight-sinusoidal trajectory.

Figure 4.17: Qualitative results for the estimation of robot's poses and landmark positions for simulated experiments. The ground truth trajectory ${}^{w}\mathbf{T}_{r_{1:N}}$ is shown in green with initial guess ${}^{w}\mathbf{T}^{(0)}_{r_{1:N}}$ in red, ground truth landmark positions ${}^{w}\boldsymbol{\ell}$ in green circles with initial guess ${}^{w}\boldsymbol{\ell}^{(0)}$ in red circles, trajectory estimate ${}^{w}\widehat{\mathbf{T}}_{r_{1:N}}$ in blue, and landmark positions estimate ${}^{w}\widehat{\boldsymbol{\ell}}$ in blue circles. The experiments are representative for a partially observable scenario (Fig. 4.17a), a fully observable scenario (Fig. 4.17b), and a combination of partially and fully observable scenario (Fig. 4.17c).

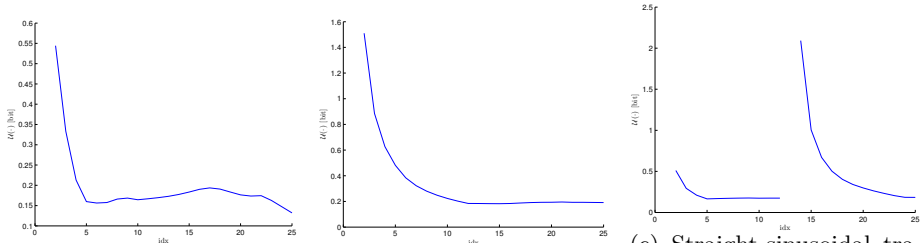


(a) ${}^{r}r_1^{rl}$ - straight (b) ${}^{r}r_2^{rl}$ - straight (c) ${}^{r}\varphi_l$ - straight

(d) ${}^{r}r_1^{rl}$ - sinusoidal (e) ${}^{r}r_2^{rl}$ - sinusoidal (f) ${}^{r}\varphi_l$ - sinusoidal

(g) ${}^{r}r_1^{rl}$ - combined (h) ${}^{r}r_2^{rl}$ - combined (i) ${}^{r}\varphi_l$ - combined

Figure 4.18: Evolution of the calibration parameter estimates ${}^{r}\widehat{\mathbf{T}}_l = ({}^{r}\widehat{r}_1^{rl}, {}^{r}\widehat{r}_1^{rl}, {}^{r}\widehat{\varphi}_l)$ (red) as a function of the incoming batches for the simulated experiments. The ground truth ${}^{r}\mathbf{T}_l$ is shown in green. The top row illustrates a partially observable scenario, the middle row a fully observable scenario, and the bottom row a combination of partially and fully observable scenario.



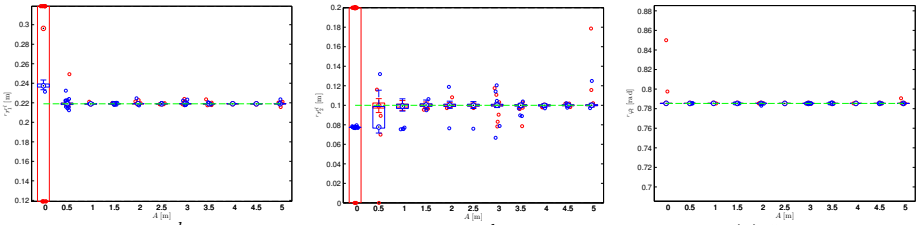
(a) Straight trajectory. (b) Sinusoidal trajectory. (c) Straight-sinusoidal trajectory.

Figure 4.19: Evolution of the information gain with respect to incoming batches. Fig. 4.19a shows a partially observable scenario, Fig. 4.19b a fully observable scenario, and Fig. 4.19c a combination of partially and fully observable scenario.

Quantitative Evaluation

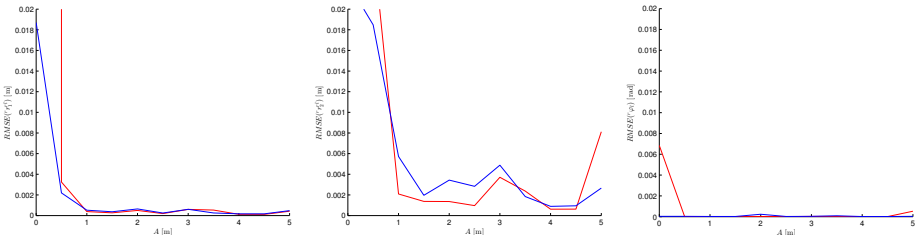
To complete the simulated experiments, we compare the performance of our algorithm against a standard non-linear least-squares (NLS) method without regularization and an extended Kalman filter (EKF). We generate a sinusoidal trajectory with varying amplitude such that it covers partial (low amplitude) to full observability (high amplitude) of the calibration parameters. To get significant statistical results, we repeat the experiment 100 times for a given amplitude and use the same initial conditions for the three methods at each run. While keeping identical calibration parameters over the entire experiment, we sample new landmark positions and orientation of the robot's trajectory for each run. We have used the algorithmic and simulation parameters outlined in Tab. 4.1 and Tab. 4.2. The sinusoid function amplitudes range from $A = 0$ [m] to $A = 5$ [m], with steps of $\Delta A = 0.5$ [m].

The estimated calibration parameters for the three methods are displayed in Fig. 4.20 and Fig. 4.22. Fig. 4.21 and Fig. 4.23 convey essentially the same information in *root mean-squared error (RMSE)* terms. While our online batch method leaves the unobservable parameters at their initial guess, the NLS method performs poorly in that situation. In fully observable conditions (from $A \approx 0.5$ [m]), the two methods yield similar results, but our online batch algorithm uses on average less than 50 percent of the available data. Even though we have set a fairly tight prior on the parameters, the EKF exhibits an instable behavior.



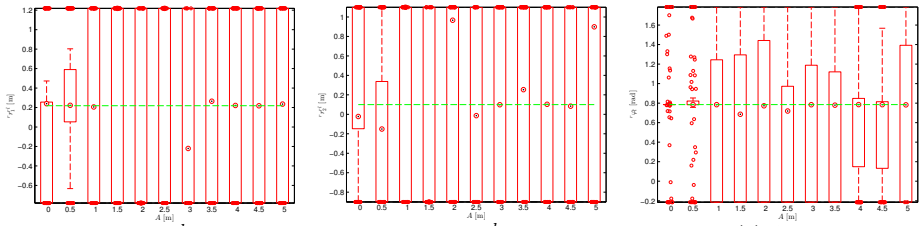
(a) ${}^{r}r_1^{rl}$. (b) ${}^{r}r_2^{rl}$. (c) ${}^{r}\varphi_l$.

Figure 4.20: Evolution of the calibration parameters at different sinusoid amplitudes for NLS (red) and our online batch algorithm (blue). The ground truth parameters are shown in the green line.



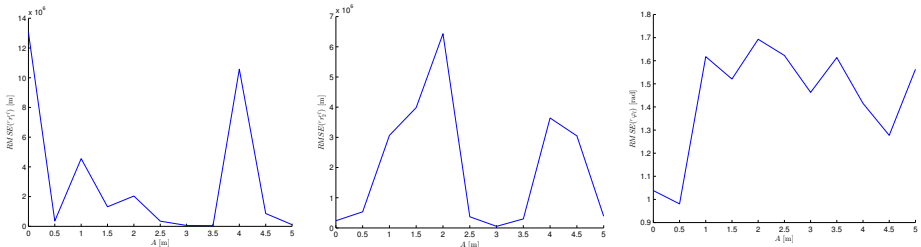
(a) ${}^{r}r_1^{rl}$. (b) ${}^{r}r_2^{rl}$. (c) ${}^{r}\varphi_l$.

Figure 4.21: RMSE of the calibration parameters at different sinusoid amplitudes for NLS (red) and our online batch algorithm (blue).



(a) ${}^{r}r_1^{rl}$. (b) ${}^{r}r_2^{rl}$. (c) ${}^{r}\varphi_l$.

Figure 4.22: Evolution of the calibration parameters at different sinusoid amplitudes for EKF. The ground truth parameters are shown in the green line.



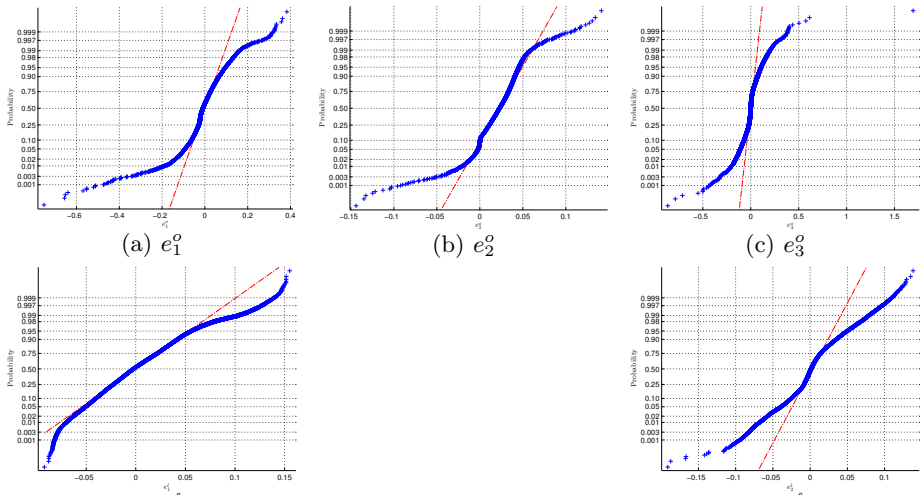
(a) ${}^{r}r_1^{rl}$. (b) ${}^{r}r_2^{rl}$. (c) ${}^{r}\varphi_l$.

Figure 4.23: RMSE of the calibration parameters at different sinusoid amplitudes for EKF.

4.2.4 Real-World Experiments

In order to validate our method on real-world data, we have used the "Lost in the Woods Dataset" provided with the courtesy of Tim Barfoot [TFB12]. This dataset contains approximately 20 minutes of a robot driving amongst a forest of tubes which serve as landmarks. The ground truth comes from a motion capture system that tracks robot motion and tube locations. For the calibration parameters, we have only access to ${}^{r}r_1^{rl} = 0.219$ [m] that was roughly measured with a tape. We assume the others are implicitly set to 0 (*i.e.*, ${}^{r}r_2^{rl} = 0$ [m] and ${}^{r}\varphi_l = 0$ [rad]). The sensor variances were estimated from part of the dataset, yielding $\sigma_v^2 = 0.0042$ [m^2/s^2], $\sigma_\omega^2 = 0.0052$ [rad^2/s^2], $\sigma_c^2 = 0.0003$ [m^2/s^2], $\sigma_r^2 = 0.0011$ [m^2], $\sigma_\varphi^2 = 0.00068$ [rad^2]. We witness in Fig. 4.24 that although the initial error terms are not perfectly normally distributed, the approximation is fair enough.

Fig. 4.25 displays the qualitative results of our online batch algorithm. Using only 25% of the measurements, we could recover accurate landmark positions, robot's poses for the informative batches, and calibration parameters. We show in Fig. 4.26 the evolution of the calibration parameters as a function of the incoming batches with a comparison to a standard nonlinear least-squares method. From this plot, we observe that the initial guess for ${}^{r}\mathbf{T}_l$ from the dataset was relatively correct. Furthermore, the only slight discrepancy (≈ 10 [mm]) between the full batch method and our algorithm appears in ${}^{r}r_2^{rl}$. This might originate from the fact that our algorithm does not use the whole dataset for estimation. Recalling Fig. 4.20, this parameter already exhibited more variability than the others, which probably stems from the lack of lateral motion of



(a) e_1^o (b) e_2^o (c) e_3^o

(d) e_1^ℓ (e) e_2^ℓ

Figure 4.24: Normal probability plots for initial odometry $\mathbf{e}^o$ (top row) and LRF $\mathbf{e}^\ell$ (bottom row) error terms.

the robot, rendering it weakly observable. The standard deviations for the estimates are 0.0017 [m] for ${}^r r_1^{rl}$, 0.0051 [m] for ${}^r r_2^{rl}$, and 0.0011 [rad] for ${}^r\varphi_l$. To conclude our real-world experiment, we display in Fig. 4.27 the evolution of the information gain as new batches are added to the online estimator. Due to the specific robot's trajectory, the information gain does not exhibit any particular spike as in Fig. 4.19c and is thus monotonously converging.

4.3 Odometry Sensors of a Consumer Car

For the second application, we investigate a calibration scenario related to an autonomous vehicle. For research purpose, we have equipped a standard Toyota Prius with a combined *global navigation satellite system (GNSS)* and *inertial navigation system (INS)*, as shown in Fig. 4.28. We have chosen the Applanix POS LV 220 (Fig. 4.29a) system that provides a highly accurate navigation solution by fusing information from two GNSS receivers, a high-end inertial measurement unit (IMU), and a rotary encoder fixed to a wheel. Through radio communication with a topographic institute providing real-time satellite corrections, we can reach centimeter-level accuracy for global positioning. In stationary cases, the rotary encoder helps stabilizing the global position. Furthermore, the IMU is supplemented by the two GNSS receivers to estimate the heading of the vehicle. The system smoothly transits into a *dead reckoning (DR)* mode in case of satellite outage. The entire POS LV system has been preliminarily calibrated using software from Applanix, such that it reaches the desired accuracy. The integrated navigation solution notably contains latitude, longitude, altitude, roll, pitch, heading, linear velocities, angular rates about the three vehicle axes, and linear accelerations. Each component of the solution comes with an associated RMSE.

The Toyota Prius itself is provided with a couple of internal sensors. For instance, the *anti-lock braking system (ABS)* uses *wheel speed sensors (WSS)*

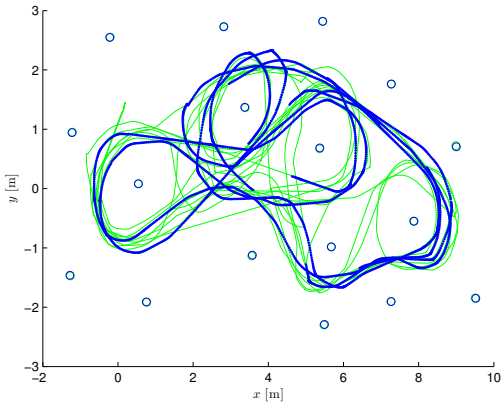
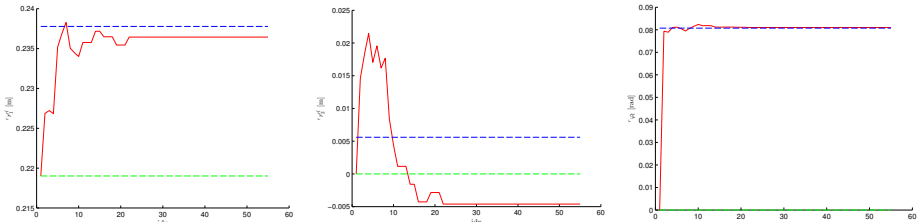


Figure 4.25: Qualitative results for the estimation of robot's poses and landmark positions for the "Lost in the Woods Dataset". The ground truth trajectory ${}^{w}\mathbf{T}_{r_{1:N}}$ is shown in green, ground truth landmark positions ${}^{w}\boldsymbol{\ell}$ in green circles, trajectory estimate ${}^{w}\widehat{\mathbf{T}}_{r_{1:N}}$ in blue, and landmark positions estimate ${}^{w}\widehat{\boldsymbol{\ell}}$ in blue circles. It can be noted that ${}^{w}\boldsymbol{\ell}$ and ${}^{w}\widehat{\boldsymbol{\ell}}$ are indistinguishable in the figure.



(a) ${}^{r}r_1^{rl}$ (b) ${}^{r}r_2^{rl}$ (c) ${}^{r}\varphi_l$

Figure 4.26: Evolution of the calibration parameter estimates ${}^{r}\widehat{\mathbf{T}}_l = ({}^{r}\widehat{r}_1^{rl}, {}^{r}\widehat{r}_1^{rl}, {}^{r}\widehat{\varphi}_l)$ (red) as a function of the incoming batches for the "Lost in the Woods Dataset". The initial guess ${}^{r}\mathbf{T}_l^{(0)}$ is shown in green and the estimates from a standard non-linear least-squares method in blue.

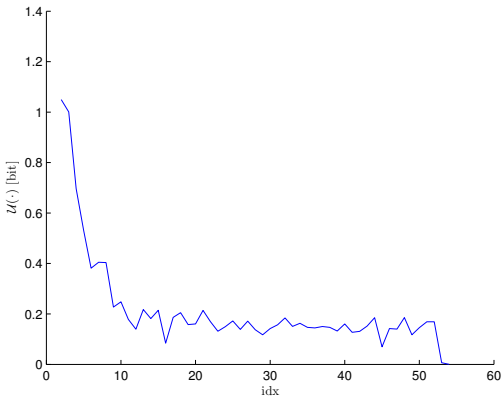


Figure 4.27: Evolution of the information gain with respect to incoming batches for the "Lost in the Woods Dataset".

on each wheel to monitor their rotational speed. The *electronic stability control (ESC)* system monitors the WSS, a *steering wheel sensor (SWS)* measuring the steering wheel angle (SWA) through potentiometers, and a lateral acceleration sensor, in order to match the driver's intention with the actual vehicle's behavior. These safety systems are embedded into *electronic control units (ECU)* that communicate with each other via a *controller area network (CAN)* interface. The CAN bus is accessible from the car interior via an *on-board diagnostics (OBD)* interface underneath the steering wheel (Fig. 4.29b). For security reasons and for keeping their industrial secrets, car manufacturers generally withhold the signification of the raw CAN messages circulating on the bus. However, by reverse engineering, we were able to identify CAN messages related to the WSS and the SWS. Our goal in this application is to build an odometry model for the car based on data from the WSS and SWS. What might seem like a trivial problem turns out to be not straightforward and offers a perfect setup for a real-world deployment of our algorithm. The first level of complexity comes from the unknown conversion factors from WSS and SWS data to exploitable measurements. Furthermore, the odometry model requires the front and rear *axle tracks*, *i.e.*, the distances between wheels on the same axle, and the *wheelbase*, *i.e.*, the distance between the rear and front axles. Our approach estimates all these quantities online, while driving the car in an arbitrary environment, by combining the CAN data with Applanix measurements.

We have further installed 4 pairs (global and rolling shutter) of digital cameras from MATRIX VISION on a roof rack, along with a Velodyne lidar HDL-64E providing 3D information of the surroundings. These sensors are not of immediate interest for the application in this section. Nevertheless, our results can favor *visual odometry* or *mapping* algorithms developed for this platform. The integration of these sensors in our calibration framework is part of future work.

Figure 4.28: The Toyota Prius for the car odometry example. Based on a GNSS/INS system from Applanix (Fig. 4.29a) and raw CAN bus data (Fig. 4.29b), we aim at calibrating the parameters of an odometry model for the vehicle.

The rest of this section shall follow a similar line as the previous one. After a brief mathematical description of the problem, we will extensively investigate the behavior of our algorithm on simulated and real-world experiments.

(a) Applanix POS LV 220.

(b) OBD interface.

Figure 4.29: Illustration of the sensors for the car odometry example. Raw CAN bus data is accessible via an OBD interface below the steering wheel and an Applanix POS LV 220 GNSS/INS provides highly accurate localization.

4.3.1 Problem Formulation

In comparison to the previous application, we need to describe relative poses and motions in a three-dimensional Euclidean space. The vehicle frame $\mathcal{F}_V$ is a right-handed Cartesian coordinate system with its first axis pointing in the forward direction of motion and aligned with a vector from the middle rear and front axles, its second axis to the left-hand side of the vehicle and aligned with the wheel axles, and its third axis up to the sky. The origin v of $\mathcal{F}_V$ is located at the middle point of a virtual rear axle, translated to the plane defined by the wheel contact points. Additionally, we need coordinate frames denoted by $\mathcal{F}_{W_{rl}}, \mathcal{F}_{W_{rr}}, \mathcal{F}_{W_{fl}}, \mathcal{F}_{W_{fr}}$ attached to each of the four wheels and with origin at the contact point of the wheels to the ground. For the odometry model, we also define a frame $\mathcal{F}_{W_v}$ attached to a virtual middle wheel, whose rotational center lies at the middle point of the front axle.

The Applanix POS LV provides the navigation solution in a reference frame $\mathcal{F}_R$ rotated by 180 degrees around the first axis of $\mathcal{F}_V$, *i.e.*, the second axis of $\mathcal{F}_R$ points to the right-hand side of the vehicle and its third axis points to the ground. The origin r of $\mathcal{F}_R$ is set at the IMU target painted on the IMU box of the POS LV system.

The navigation solution of the POS LV describes the georeferenced relative position of $\mathcal{F}_R$ with respect to a rotating *Earth-centered Earth-fixed (ECEF)* coordinate system denoted by $\mathcal{F}_E$ with origin e at the center of mass of the Earth, third axis coincident with the instantaneous Earth rotational axis and pointing to the north pole, first axis pointing to the first meridian, and second axis completing the right-handed coordinate system.

The relative orientation of $\mathcal{F}_R$ is computed with respect to a *north-east-down (NED) local-level frame (LLF)* $\mathcal{F}_L$. The origin l of $\mathcal{F}_L$ is located at the intersection of the line from e to r with the Earth's surface, determined by the WGS84 *reference ellipsoid*. Its first and second axes form a plane tangent to the local Earth's surface. The first axis points towards the north pole along the local meridian, its second axis to the east, and its third axis to the Earth's center.

We can further define a fixed *mapping frame* $\mathcal{F}_M$ as the *east-north-up (ENU)* LLF computed after the initial leveling and heading alignment of the POS LV system. The first axis of $\mathcal{F}_M$ points to the east, the second axis to the north along the local meridian, and the third axis to the sky. It can be noted that the frame $\mathcal{F}_M$ corresponds to the world frame in the application presented in Sec. 4.2, *i.e.*, relative poses and motion of the vehicle will be resolved in that frame. The ENU coordinate system is the natural representation for creating topographic maps and hence localizing our vehicle in a national map.

In this application, we shall estimate the time-varying relative pose ${}^{m}\mathbf{T}_{v}(t)$ of the vehicle frame $\mathcal{F}_V$ with respect to the fixed mapping frame $\mathcal{F}_M$, along with the time-invariant relative poses of the wheel frames $\mathcal{F}_{W_{rl}}, \mathcal{F}_{W_{rr}}, \mathcal{F}_{W_{fl}}, \mathcal{F}_{W_{fr}}, \mathcal{F}_{W_v}$ with respect to the vehicle frame $\mathcal{F}_V$, and from the Applanix frame $\mathcal{F}_R$ to the vehicle frame $\mathcal{F}_V$. Furthermore, we need scale factors $k_{rl}, k_{rl}, k_{fl}, k_{fr}$ for each WSS to properly interpret the raw CAN data. We might additionally use the Applanix's WSS that we scale with k_{dmi}. Finally, we map SWS measurements to steering angles of the virtual middle wheel with a third-order polynomial parameterized by the scalars $a_{0:3}$. The configuration of the coordinate frames is depicted in Fig. 4.30.

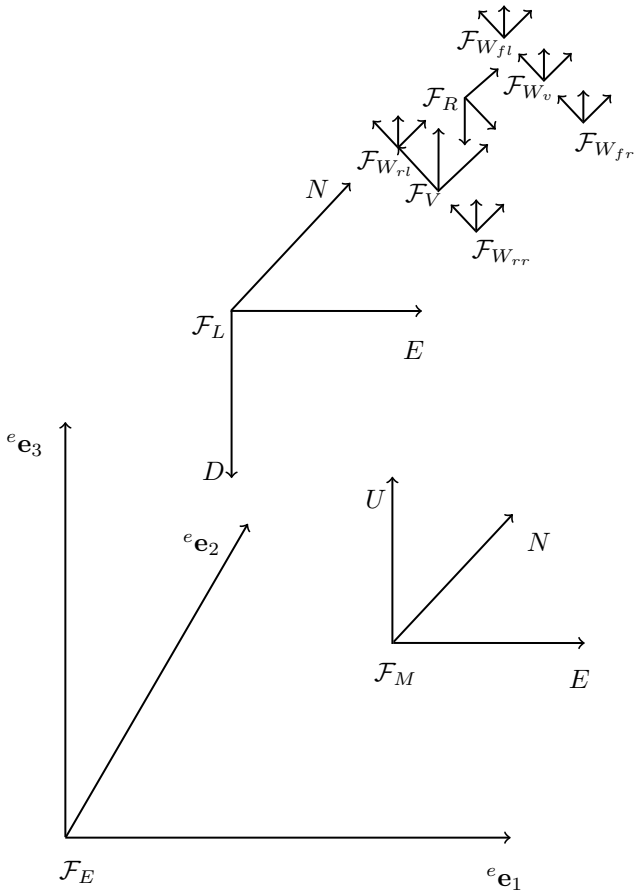


Figure 4.30: Setup of the coordinate frames for the car odometry example.

Motion Model

The motion of the vehicle can be characterized by the time-varying transformation ${}^{m}\mathbf{T}_{v}(t)$, which we decompose into a time-varying rotation matrix ${}^{m}\mathbf{R}_{v}(t)$ and translation vector ${}^{m}\mathbf{r}^{mv}(t)$. The motion of $\mathcal{F}_V$ can therefore be described by the following differential equations

$$\begin{aligned} {}^{m}\dot{\mathbf{r}}^{mv}(t) &= {}^{m}\mathbf{v}^{mv}(t), \\ {}^{m}\dot{\mathbf{R}}_{v}(t) &= \left[{}^{m}\boldsymbol{\omega}^{mv}\right]_{\times}(t)\ {}^{m}\mathbf{R}_{v}(t). \end{aligned} \tag{4.33}$$

The Applanix POS LV provides an integrated navigation solution at 100 Hz. Consequently, if we were to adopt a time-discretized motion model as in Sec. 4.2, the estimation problem would rapidly become intractable. Moreover, we would require interpolation between poses to build error terms for odometry measurements. We therefore borrow the technique presented in [FBS12]. The time-varying relative pose ${}^{m}\mathbf{T}_{v}(t)$ is approximated by a finite set of temporal B-spline basis functions. The relative pose and motion can be evaluated at any discrete time with a linear combination of basis functions. Under that paradigm, the estimation problem amounts to determining a relatively small set of linear coefficients. We refer to the aforementioned publication for a more in-depth treatment. For the sake of clarity, we henceforth assume that our continuous-time motion model is encoded with a B-spline representation and that we can query it, along with its time derivatives, at any given discrete time.

Following on the motion model in Sec. 4.2, the car cannot move unconstrained in space. Hence, we shall add a similar non-holonomic constraint expressed as

$${}^{m}\mathbf{v}^{mv}(t) = \left[{}^{m}\boldsymbol{\omega}^{mv}\right]_{\times}(t)\ {}^{m}\mathbf{r}^{mv}(t). \tag{4.34}$$

Navigation Solution Model

From the *geodetic coordinates* and the WGS84 datum, we can construct the fixed vector ${}^{e}\mathbf{r}^{em}$ from the origin of $\mathcal{F}_E$ to the origin of $\mathcal{F}_M$, resolved in $\mathcal{F}_E$. This vector is computed only once, after initial convergence of the navigation solution, and will define the anchor for the mapping frame $\mathcal{F}_M$. Similarly, we can initially compute the rotation matrix ${}^{m}\mathbf{R}_{e}$ that transforms vectors from $\mathcal{F}_E$ to $\mathcal{F}_M$. After this operation, our mapping frame is entirely georeferenced.

The Applanix POS LV system outputs navigation measurements at discrete timesteps k that we shall relate to the latent variables ${}^{m}\mathbf{T}_{v}(k)$ and ${}^{v}\mathbf{T}_{r}$. We can interpret a navigation measurement at time k as a position vector ${}^{e}\mathbf{r}_{k}^{er}$, from Earth's center to Applanix reference frame, and a rotation matrix ${}^{l}\mathbf{R}_{r_k}$, from Applanix reference frame to the local-level frame. We can first construct the rotation matrix ${}^{e}\mathbf{R}_{l_k}$ from the geodetic coordinates and apply a composition of affine transformations to get the inverse model for ${}^{m}\mathbf{T}_{v}(k)$ as

$$\begin{aligned} {}^{m}\mathbf{R}_{v}(k) &= {}^{m}\mathbf{R}_{e}\ {}^{e}\mathbf{R}_{l_k}\ {}^{l}\mathbf{R}_{r_k}\ {}^{v}\mathbf{R}_{r}^{T}, \\ {}^{m}\mathbf{r}^{mv}(k) &= {}^{m}\mathbf{R}_{e}\left({}^{e}\mathbf{r}_{k}^{er} - {}^{e}\mathbf{r}^{em}\right) - {}^{m}\mathbf{R}_{v}(k)\ {}^{v}\mathbf{r}^{vr}. \end{aligned} \tag{4.35}$$

Similarly, we can express the forward model $h_r({}^m\mathbf{T}_v(k),\ {}^v\mathbf{T}_r)$ for the Applanix measurements ${}^e\mathbf{r}_k^{er}$ and $({}^e\mathbf{R}_{l_k}, {}^l\mathbf{R}_{r_k})$ as

$$\begin{aligned} {}^e\mathbf{r}_k^{er} &= {}^m\mathbf{R}_e^T\ ({}^m\mathbf{r}^{mv}(k) + {}^m\mathbf{R}_v(k)\ {}^v\mathbf{r}^{vr}) + {}^e\mathbf{r}^{em}, \\ {}^e\mathbf{R}_{l_k}\ {}^l\mathbf{R}_{r_k} &= {}^m\mathbf{R}_e^T\ {}^m\mathbf{R}_v(k)\ {}^v\mathbf{R}_r. \end{aligned} \tag{4.36}$$

Odometry Model

Based on the reverse engineering of the CAN messages, we have identified a message WSS_r related to the linear velocities of the rear wheels, a message WSS_f related to the linear velocities of the front wheels, and a message SWS related to the steering wheel angle. As each message arrives asynchronously, we shall process them individually. In the following, we take inspiration on the elegant vector algebra formulation presented in [KS12a]. We first recall here that ${}^m\mathbf{T}_v(t)$, ${}^m\mathbf{v}^{mv}(t)$, and ${}^m\boldsymbol{\omega}^{mv}(t)$, represent our nuisance latent variables. Apart from ${}^v\mathbf{T}_r$, the set of calibration latent variables contains the transformations ${}^v\mathbf{T}_{w_{rl}}$, ${}^v\mathbf{T}_{w_{rr}}$, ${}^v\mathbf{T}_{w_{fl}}$, ${}^v\mathbf{T}_{w_{fr}}$, ${}^v\mathbf{T}_{w_v}$, the scale factors $k_{rl}, k_{rl}, k_{fl}, k_{fr}$ and the polynomial coefficients $a_{0:3}$.

For the WSS_r, we receive a measurement $\mathbf{v}_k^r = (v_{l_k}^r, v_{r_k}^r)^T$ at time k, $v_{l_k}^r$ and $v_{r_k}^r$ representing velocity measurements for the left and right rear wheels respectively. Using Equ. 4.10, we can express the linear velocities at time k of the rear wheel frames $\mathcal{F}_{W_{rl}}$ and $\mathcal{F}_{W_{rr}}$ with respect to $\mathcal{F}_M$ and resolved in $\mathcal{F}_M$ as

$$\begin{aligned} {}^m\mathbf{v}^{mw_{rl}}(k) &= {}^m\mathbf{v}^{mv}(k) + {}^m\boldsymbol{\omega}^{mv}(k) \times {}^m\mathbf{r}^{vw_{rl}}, \\ {}^m\mathbf{v}^{mw_{rr}}(k) &= {}^m\mathbf{v}^{mv}(k) + {}^m\boldsymbol{\omega}^{mv}(k) \times {}^m\mathbf{r}^{vw_{rr}}. \end{aligned} \tag{4.37}$$

We can further express Equ. 4.37 in $\mathcal{F}_V$ as

$$\begin{aligned} {}^v\mathbf{v}^{mw_{rl}}(k) &= {}^m\mathbf{R}_v^T(k)\ {}^m\mathbf{v}^{mv}(k) + {}^m\mathbf{R}_v^T(k)\ {}^m\boldsymbol{\omega}^{mv}(k) \times {}^v\mathbf{r}^{vw_{rl}}, \\ {}^v\mathbf{v}^{mw_{rr}}(k) &= {}^m\mathbf{R}_v^T(k)\ {}^m\mathbf{v}^{mv}(k) + {}^m\mathbf{R}_v^T(k)\ {}^m\boldsymbol{\omega}^{mv}(k) \times {}^v\mathbf{r}^{vw_{rr}}. \end{aligned} \tag{4.38}$$

Finally, we transform the linear velocities in $\mathcal{F}_{W_{rl}}$ and $\mathcal{F}_{W_{rr}}$.

$$\begin{aligned} {}^{w_{rl}}\mathbf{v}^{mw_{rl}}(k) &= {}^v\mathbf{R}_{w_{rl}}^T\ {}^v\mathbf{v}^{mw_{rl}}(k), \\ {}^{w_{rl}}\mathbf{v}^{mw_{rr}}(k) &= {}^v\mathbf{R}_{w_{rr}}^T\ {}^v\mathbf{v}^{mw_{rr}}(k). \end{aligned} \tag{4.39}$$

We assume that the rear wheel frames always remain aligned with the vehicle frame and we can thus build the forward models $h_{w_{rl}}(\cdot)$ and $h_{w_{rr}}(\cdot)$ from Equ. 4.39 as

$$\begin{aligned} &h_{w_{rl}}({}^m\mathbf{T}_v(k),\ {}^m\mathbf{v}^{mv}(k),\ {}^m\boldsymbol{\omega}^{mv}(k),\ {}^v\mathbf{T}_{w_{rl}},\ k_{rl}) \\ &\quad = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} k_{rl}\ {}^{w_{rl}}\mathbf{v}^{mw_{rl}}(k) \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T, \\ &h_{w_{rr}}({}^m\mathbf{T}_v(k),\ {}^m\mathbf{v}^{mv}(k),\ {}^m\boldsymbol{\omega}^{mv}(k),\ {}^v\mathbf{T}_{w_{rr}},\ k_{rr}) \\ &\quad = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} k_{rr}\ {}^{w_{rr}}\mathbf{v}^{mw_{rr}}(k) \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T. \end{aligned} \tag{4.40}$$

We adopt here a similar strategy as in Sec. 4.2 to account for the wheel slip constraints. Furthermore, we assume that the linear velocity at the wheel contact point is tangential to the local plane and therefore constrain the third component of the velocity to zero with a pseudo measurement. As earlier, the forward models consist in a composition of affine transformations from ${}^{m}\mathbf{v}^{mv}(k)$ to ${}^{w_{rl/r}}\mathbf{v}^{mw_{rl/r}}(k)$. The scaling factors k_{rl}, k_{rr} convert raw CAN data to linear velocities. We also note that the inverse model, *i.e.*, going from the measurements to the parameters, corresponds to the standard differential-drive kinematics in a two-dimensional space.

For the SWS, we consider a typical model for car-like vehicles, namely the *bicycle model*. We receive a measurement $\widetilde{\varphi}_k$ at time k that we want to relate to the steering angle φ_k of a virtual middle wheel on the front axle. Using the above derivations, we can compute the linear velocity of $\mathcal{F}_{W_v}$ with respect to $\mathcal{F}_M$ and resolved in $\mathcal{F}_{W_v}$ as

$$ {}^{w_v}\mathbf{v}^{mw_v}(k) \quad = \quad {}^{v}\mathbf{R}_{w_v}^{T} \; {}^{v}\mathbf{v}^{mw_v}(k). \tag{4.41} $$

From the car model, we can assume that ${}^{v}\mathbf{R}_{w_v}^{T}$ always represents a rotation of an angle φ_k around the third axis of $\mathcal{F}_{W_v}$. Furthermore, we enforce that the wheel is not slipping laterally, *i.e.*, the second component of ${}^{w_v}\mathbf{v}^{mw_v}(k)$ vanishes. If we express ${}^{w_v}\mathbf{v}^{mw_v}(k) = ({}^{w_w}v_1^{mw_v}(k),\ {}^{w_w}v_2^{mw_v}(k),\ {}^{w_w}v_3^{mw_v}(k))^T$ and solve for the slipping constraint, we obtain

$$ \varphi_k \quad = \quad \text{atan2}({}^{w_w}v_2^{mw_v}(k),\ {}^{w_w}v_1^{mw_v}(k)). \tag{4.42} $$

It is worth noting that Equ. 4.42 corresponds to the geometric derivations of [BBeM01] in the two-dimensional space. However, we believe that our formulation is more generic. From φ_k, we can express the forward model $h_{w_v}(\cdot)$ for $\widetilde{\varphi}_k$ as

$$ h_{w_v}({}^{m}\mathbf{T}_v(k),\ {}^{m}\mathbf{v}^{mv}(k),\ {}^{m}\boldsymbol{\omega}^{mv}(k),\ {}^{v}\mathbf{T}_{w_{rl}}, a_{0:3}) \quad = \quad \sum_{i=0}^{3} a_i \varphi_k^i. \tag{4.43} $$

Measurements $\widetilde{\varphi}_k$ alone are not sufficient to generate an inverse model for vehicle's poses. Furthermore, we remark that Equ. 4.42 introduces singularities. Although we can measure $\widetilde{\varphi}_k$, the angle φ_k is not defined for ${}^{w_v}\mathbf{v}^{mw_v}(k) = \mathbf{0}$. We can nevertheless handle this case by discarding the corresponding measurements.

For the WSS$_f$, we acquire a measurement $\mathbf{v}_k^f = (v_{l_k}^f, v_{r_k}^f)^T$ at time k, $v_{l_k}^f$ and $v_{r_k}^f$ representing velocity measurements for the left and right front wheels respectively. The forward model for $\mathbf{v}_k^f$ is a combination of Equ. 4.39 and Equ. 4.42. We first express the linear velocities in the wheel frames as

$$ \begin{aligned} {}^{w_{fl}}\mathbf{v}^{mw_{fl}}(k) \quad &= \quad {}^{v}\mathbf{R}_{w_{fl}}^{T} \; {}^{v}\mathbf{v}^{mw_{fl}}(k), \\ {}^{w_{fr}}\mathbf{v}^{mw_{fr}}(k) \quad &= \quad {}^{v}\mathbf{R}_{w_{fr}}^{T} \; {}^{v}\mathbf{v}^{mw_{fr}}(k), \end{aligned} \tag{4.44} $$

solve for the rotation angles φ_k^{fl} and φ_k^{fr} around the third axis of $\mathcal{F}_{W_{fl/fr}}$ such that the slipping constraints are satisfied, and insert the resulting angles into ${}^{v}\mathbf{R}_{w_{fl}}^{T}$ and ${}^{v}\mathbf{R}_{w_{fr}}^{T}$. Finally, the forward model can be expressed as

$$\begin{aligned} & h_{w_{fl}}({}^{m}\mathbf{T}_{v}(k),\ {}^{m}\mathbf{v}^{mv}(k),\ {}^{m}\boldsymbol{\omega}^{mv}(k),\ {}^{v}\mathbf{T}_{w_{fl}},\ k_{fl}) \\ & \quad = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} k_{fl}\ {}^{w_{fl}}\mathbf{v}^{mw_{fl}}(k) \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^{T}, \\ & h_{w_{fr}}({}^{m}\mathbf{T}_{v}(k),\ {}^{m}\mathbf{v}^{mv}(k),\ {}^{m}\boldsymbol{\omega}^{mv}(k),\ {}^{v}\mathbf{T}_{w_{fr}},\ k_{fr}) \\ & \quad = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} k_{fr}\ {}^{w_{fr}}\mathbf{v}^{mw_{fr}}(k) \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^{T}. \end{aligned} \tag{4.45}$$

As for $\widetilde{\varphi}_k$, measurements $\mathbf{v}_k^f$ alone are insufficient to build an inverse model for vehicle's poses.

The forward model $h_{w_{dmi}}(\cdot)$ for the Applanix's WSS, hereafter referred to as *distance measurement indicator (DMI)*, is assumed identical to $h_{w_{rr}}(\cdot)$ since the encoders are collocated, *i.e.*, we predict a measurement v_k^{dmi} with

$$\begin{aligned} & h_{w_{dmi}}({}^{m}\mathbf{T}_{v}(k),\ {}^{m}\mathbf{v}^{mv}(k),\ {}^{m}\boldsymbol{\omega}^{mv}(k),\ {}^{v}\mathbf{T}_{w_{fr}},\ k_{dmi}) \\ & \quad = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} k_{dmi}\ {}^{w_{fr}}\mathbf{v}^{mw_{fr}}(k) \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^{T}. \end{aligned} \tag{4.46}$$

We might additionally define a pseudo forward model that constrains angular velocity in the vehicle frame. We express this model $h_c(\cdot)$ as

$$h_c({}^{m}\mathbf{T}_{v}(k),\ {}^{m}\boldsymbol{\omega}^{mv}(k)) \quad = \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} {}^{m}\mathbf{R}_{v}^{T}(k)\ {}^{m}\boldsymbol{\omega}^{mv}(k). \tag{4.47}$$

If we synchronize the set of odometry measurements $\{\mathbf{v}_k^r, \mathbf{v}_k^f, v_k^{dmi}, \widetilde{\varphi}_k\}$, *e.g.*, by interpolation, we can construct a higher-fidelity inverse model for predicting vehicle's linear and angular velocities in a two-dimensional space. The steering wheel angle $\widetilde{\varphi}_k$ can be related to the steering angle φ_k of the virtual middle wheel through Equ. 4.43. In a second step, the *Ackermann steering geometry* provides the steering angles φ_k^{fl} and φ_k^{fr} of the left and front right wheels respectively. From there, we can establish an overdetermined system of linear equations by stacking the measurements on the right-hand side and computing the least-squares estimate for the first component of ${}^{v}\mathbf{v}^{mv}(k)$ and the third component of ${}^{v}\boldsymbol{\omega}^{mv}(k)$. Finally, these estimates are used to integrate the time-discretized motion model in Equ. 4.20. Alternatively, an extended Kalman filter could integrate these measurements while retaining their uncertainty [BBeM01].

We are aware that the presented odometry model is only a rough approximation of the underlying physical system. In reality, there is a lever arm between the steering points of the front wheels and their contact points. Furthermore, a car is not a rigid body; we have neglected here the effect of the suspension system or tire deformations. Finally, during drive, car wheels might get misaligned with respect to the vehicle frame, *i.e.*, $\mathcal{F}_{W_i}$ can have a rolling angle with respect to $\mathcal{F}_V$. In the current setup, we have no means to estimate these effects. Nevertheless, as will be demonstrated in this section, we were able to obtain relevant results.

Estimation Model

Before diving into the probabilistic model, it is worth recapitulating the variables of the system. With respect to our calibration algorithm, the nuisance variable is $\mathbf{\Psi} = \{{}^{m}\mathbf{T}_{v}(t)\}$, defining the time-varying vehicle's relative pose to a fixed mapping frame. The dimensionality of $\mathbf{\Psi}$ is bounded by a chosen spline order s^{o} and knots number s^{k}. Initial guesses for the B-spline coefficients are calculated by fitting the inverse model in Equ. 4.35 with Applanix navigation solution measurements. The calibration variable of interest is $\mathbf{\Theta} = \{k_{rl}, k_{rr}, k_{fl}, k_{fr}, k_{dmi}, a_{0:3}, {}^{v}\mathbf{T}_{r}, {}^{v}\mathbf{T}_{w_{rl}}, {}^{v}\mathbf{T}_{w_{rr}}, {}^{v}\mathbf{T}_{w_{fl}}, {}^{v}\mathbf{T}_{w_{fr}}, {}^{v}\mathbf{T}_{w_{v}}\}$. The transformations between the wheels and the vehicle frame can be further simplified if we take into account the vehicle's symmetry and our assumptions. First, the rear wheels remain aligned with the vehicle frame, *i.e.*,

$$\begin{aligned} {}^{v}\mathbf{R}_{w_{rl}} &= \mathbf{I}, \\ {}^{v}\mathbf{R}_{w_{rr}} &= \mathbf{I}. \end{aligned} \tag{4.48}$$

The rotation matrices ${}^{v}\mathbf{R}_{w_{fl}}$, ${}^{v}\mathbf{R}_{w_{fr}}$, and ${}^{v}\mathbf{R}_{w_{w}}$ are determined by the time-derivative of ${}^{m}\mathbf{T}_{v}(t)$ and the lateral slip constraints, assuming rotation only around the third axis of $\mathcal{F}_{W_{fl/fr/w}}$. For the translation vectors, we have the following relationships

$$\begin{aligned} {}^{v}\mathbf{r}^{vw_{rl}} &= -{}^{v}\mathbf{r}^{vw_{rr}}, \\ {}^{v}\mathbf{r}^{vw_{fl}} &= -{}^{v}\mathbf{r}^{vw_{fr}}, \\ {}^{v}\mathbf{r}^{vw_{fl}} &= {}^{v}\mathbf{r}^{vw_{v}} + {}^{v}\mathbf{r}^{w_{v}w_{fl}}, \\ {}^{v}\mathbf{r}^{vw_{fr}} &= {}^{v}\mathbf{r}^{vw_{v}} - {}^{v}\mathbf{r}^{w_{v}w_{fl}}, \end{aligned} \tag{4.49}$$

By setting the origins of the wheels and vehicle frames in the same plane, we can further reduce the number of degrees of freedom. The translation vectors can be fully described by the three scalars e_{r}, e_{f}, and L, such that

$$\begin{aligned} {}^{v}\mathbf{r}^{vw_{rl}} &= \begin{pmatrix} 0 & e_{r} & 0 \end{pmatrix}^{T}, \\ {}^{v}\mathbf{r}^{w_{v}w_{fl}} &= \begin{pmatrix} 0 & e_{f} & 0 \end{pmatrix}^{T}, \\ {}^{v}\mathbf{r}^{vw_{v}} &= \begin{pmatrix} L & 0 & 0 \end{pmatrix}^{T}. \end{aligned} \tag{4.50}$$

Initial guesses for e_{r} (rear track), e_{f} (front track), and L (wheelbase), are available from the datasheet of the Toyota Prius. The transformation ${}^{v}\mathbf{T}_{r}$, from the Applanix reference frame to the vehicle frame, has six degrees of freedom, three for ${}^{v}\mathbf{r}^{vr}$ and three for ${}^{v}\mathbf{R}_{r}$. We obtain an initial guess for ${}^{v}\mathbf{r}^{vr}$ with a measuring tape. Although $\mathcal{F}_{R}$ should be by design rotated of a 180 degrees angle around the first axis of $\mathcal{F}_{V}$, we leave ${}^{v}\mathbf{R}_{r}$ unconstrained for the sake of this application and use this information as initial guess. So, all in all, $\mathbf{\Theta}$ is an 18-dimensional random vector.

To construct our estimation problem, we shall define an error term relating each measurement to its forward model, *i.e*,

$$\begin{aligned} \mathbf{e}_k^r &= {}^e\mathbf{T}_{r_k} - h_r(\cdot), \\ \mathbf{e}_k^{w_{rl/rr/fl/fr/dmi}} &= \mathbf{v}_k^{w_{rl/rr/fl/fr/dmi}} - h_{w_{rl/rr/fl/fr/dmi}}(\cdot), \\ e^{w_w} &= \widetilde{\varphi}_k - h_{w_w}(\cdot), \end{aligned} \tag{4.51}$$

with the linear velocity measurements containing zeros in the second and third components. We can further add a pseudo error term at each spline knot, for the rotational constraints of the vehicle frame, defined as

$$\mathbf{e}_k^c = \mathbf{0} - h_c(\cdot). \tag{4.52}$$

As in Sec. 4.2, the error terms are normally distributed. The covariance matrix $\mathbf{\Sigma}_k^r$ for $\mathbf{e}_k^r$ can be inferred from the RMS errors reported by the Applanix POS LV. For each WSS and for the SWS, we estimate a sample variance from a training dataset, yielding $\sigma^2_{w_{rl/rr/fl/fr/dmi}}$ and $\sigma^2_{w_v}$. From the training dataset, we can also approximate variances for the system constraints. We denote them by σ_l^2 for a lateral constraint, σ_n^2 for a terrain normal constraint, $\sigma^2_{\omega_x}$ and $\sigma^2_{\omega_y}$ for the rotational constraints. It is worth noting here that the odometry variances, along with the constraint variances, should compensate for the model deficiency and minimize the effects of the suspension system, tire deformations, and wheel misalignments.

The Jacobian matrices for the error terms are calculated analytically using the chain rule for derivatives, the top-level functions being simple polynomials. We linearize affine transformations using the derivations in [Fur11]. Finally, as the sparse structure of the joint probability density function is clearly reflected from the arguments to the different forward models, we skip the graphical model representation.

4.3.2 Observability Analysis

As in Sec. 4.2.2, an a priori analysis provides a valuable starting point for understanding the estimation model and identifying the problematic situations. The calibration parameter space being higher dimensional, we restrict our attention to the forward model and the Jacobian matrix structure.

Joint Forward Model

We assume a data sample $\mathcal{D} = \{\mathbf{v}^r_{1:N}, \mathbf{v}^f_{1:N}, \widetilde{\varphi}_{1:N}, \mathbf{v}^{dmi}_{1:N}, {}^e\mathbf{T}_{r_{1:N}}\}$, containing all the sensor outputs for a time span of size N. The data sample $\mathcal{D}$ can be generated by a set of poses ${}^m\mathbf{T}_{v_{1:N}}$, linear and angular velocities ${}^m\mathbf{v}^{mv}_{1:N}$ and ${}^m\boldsymbol{\omega}^{mv}_{1:N}$, a relative pose ${}^v\mathbf{T}_r$ between the vehicle and Applanix frames, translation vectors from the vehicle to the wheel frames ${}^v\mathbf{r}^{vw_{rl}}$, ${}^v\mathbf{r}^{w_v w_{fl}}$, and ${}^v\mathbf{r}^{vw_v}$, and intrinsic odometry parameters k_{rl}, k_{rr}, k_{fl}, k_{fr}, k_{dmi}, and $a_{0:3}$. Constant to the model, we additionally have the relative pose ${}^e\mathbf{T}_m$ between the mapping and the Earth frames. Here, without loss of generality, we have simplified the analysis by enforcing synchronous data for each sensor and by considering a time-discretized motion model. As earlier, we investigate the properties of the joint forward model

$$\mathcal{D} \quad = \quad H(\boldsymbol{\psi}, \boldsymbol{\theta}), \tag{4.53}$$

where we have, for the sake of clarity, gathered the different variables into calibration and nuisance variables. The goal of the estimation process is to determine the inverse function $H^{-1}(\mathcal{D})$ and we seek here the conditions under which this is feasible. As in Sec. 4.2.2, we shall perform an informal analysis and confirm our thoughts through the Jacobian matrix structure. A traditional observability analysis is not required as our algorithm is able to handle it online. We aim here only at establishing some unobservable configurations in order to have a finer comprehension of the problem.

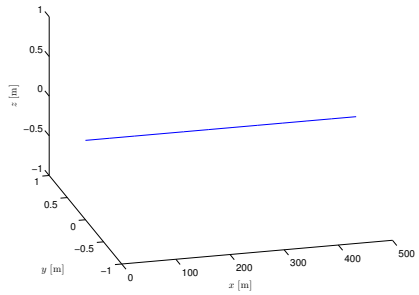
The problem at hand is in essence closely related to the previous application with the Applanix sensor playing the role of the laser rangefinder, except that the landmark positions are not part of the latent variables. We can start our analysis by considering the odometry subsystem. Having enforced the wheel constraints and skipped the trivial case of zero motion, any change in the wheel scale factors results in a distinct output of the forward model. Consequently, $\{k_{rl}, k_{rr}, k_{fl}, k_{fr}, k_{dmi}\}$ will always remain fully observable. If the vehicle is never turning in its local tangent plane, the four wheels will have the same linear velocity and changing any of the translation vectors $\{{}^{v}\mathbf{r}^{vw_{rl}}, {}^{v}\mathbf{r}^{w_v w_{fl}}, {}^{v}\mathbf{r}^{vw_v}\}$ will have no effect on the sensor output. Similarly, the polynomial coefficients $a_{1:3}$ can also be arbitrary in that case since the steering angle of the virtual wheel is zero.

The remaining calibration parameter ${}^{v}\mathbf{T}_r$, composed of ${}^{v}\mathbf{R}_r$ and ${}^{v}\mathbf{r}^{vr}$, links the vehicle and the Applanix frames. When the car drives in a plane and we apply a translation along the normal direction to ${}^{m}\mathbf{T}_{v_{1:N}}$ and ${}^{v}\mathbf{r}^{vr}$ simultaneously, none of the outputs of $H(\boldsymbol{\psi}, \boldsymbol{\theta})$ shall be modified. Consequently, the car must drive on a non-planar trajectory to observe the third component of the translation ${}^{v}\mathbf{r}^{vr}$.

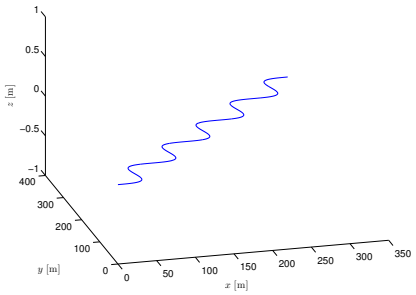
Jacobian Matrix Structure

Continuing on the observability analysis, we shall examine how the structure of the Jacobian matrix reflects our intuitions and focus on the scenes depicted in Fig. 4.31. These setups shall cover most of the relevant situations, ranging from partial to full observability.

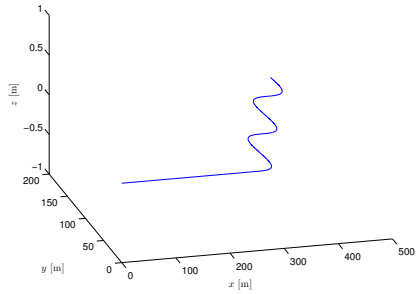
We have summarized in Fig. 4.32 the behavior of the singular values for the trajectories in Fig. 4.31a, Fig. 4.31b, Fig. 4.31d, and Fig. 4.31e. For all these configurations, the Jacobian matrix $\mathbf{J}_{\boldsymbol{\psi}}$ of the nuisance variables, *i.e.*, ${}^{m}\mathbf{T}_{v_{1:N}}$, has full rank since the pose measurements are georeferenced. The rank deficiency is therefore entirely caused by the calibration variable $\boldsymbol{\theta}$. For the straight-planar path, we witness, as expected, that 6 singular values corresponding to the 5 scale factors and the first coefficient of the steering wheel transmission are well-distinguishable from zero. In the sinusoidal-planar path, we have 6 additional singular values clearly departing from zero, corresponding to the axle tracks, the wheel base, and the remaining coefficients of the steering wheel transmission. Besides, 5 other singular values linked to ${}^{v}\mathbf{T}_r$ are slightly deviating from zero. All in all, only one singular value, associated with the third component of ${}^{v}\mathbf{r}^{vr}$ can definitely be considered as zero. Nevertheless, the other components of the transformation from the Applanix reference frame $\mathcal{F}_R$ to the



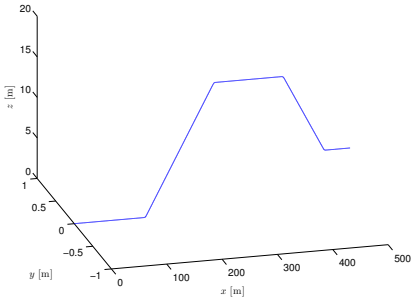
(a) Straight-planar trajectory.



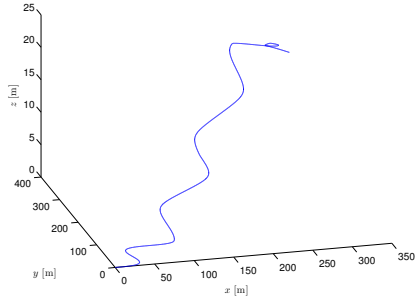
(b) Sinusoidal-planar trajectory.



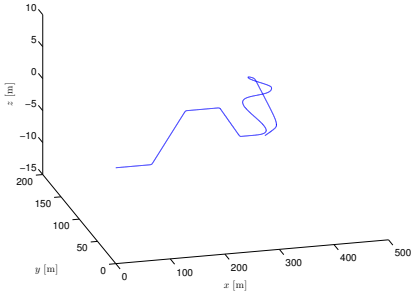
(c) Straight-sinusoidal-planar trajectory.



(d) Straight-hill trajectory.



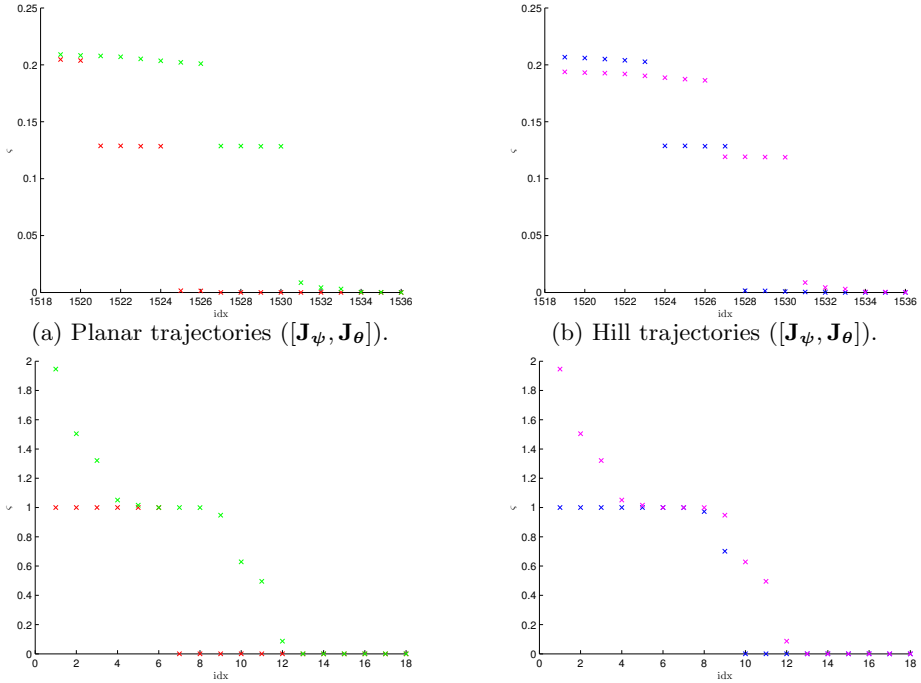
(e) Sinusoidal-hill trajectory.



(f) Straight-sinusoidal-hill trajectory.

Figure 4.31: Examples of car's trajectories. The blue line corresponds to the car's path ${}^{m}\mathbf{T}_{v_{1:N}}$. These setups cover the different observability cases, ranging from partial (Fig. 4.31a, Fig. 4.31b, Fig. 4.31c, and Fig. 4.31d) to full observability (Fig. 4.31e and Fig. 4.31f). The plots are ordered from top to bottom and left to right by increasing degree of observability of the calibration parameters.

vehicle frame $\mathcal{F}_V$ shall be weakly observable, as compared to the rest of the calibration parameters. When the car follows a straight trajectory over a hill, 3 more singular values are significantly above zero, linked to two components of ${}^v\mathbf{R}_r$ and the third component of ${}^v\mathbf{r}^{vr}$. Finally, for the sinusoidal path over a hill, the rest of the singular values become nonzero. As before, it is worth noting that the singular values associated with the transformation ${}^v\mathbf{T}_r$ have much smaller magnitude than the others.



(a) Planar trajectories ($[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]$).

(b) Hill trajectories ($[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]$).

(c) Planar trajectories ($\mathbf{A}_{\boldsymbol{\theta}}$).

(d) Hill trajectories ($\mathbf{A}_{\boldsymbol{\theta}}$).

Figure 4.32: The 18 smallest singular values of the scaled $[\mathbf{J}_{\boldsymbol{\psi}}, \mathbf{J}_{\boldsymbol{\theta}}]$ (top row) and $\mathbf{A}_{\boldsymbol{\theta}}$ (bottom row) for the straight-planar (red), sinusoidal-planar (green), straight-hill (blue), and sinusoidal-hill (magenta) trajectory.

4.3.3 Simulated Experiments

Having largely demonstrated the sanity of our algorithm in Sec. 4.2.3, we shall directly focus on its overall behavior when applied to the aforementioned scenarios, especially to the partially observable configurations. Apart from the ground truth, a simulation offers a finer control on the experimental conditions and the possibility to test at no cost any vehicle's trajectory.

In the following simulations, we have evaluated our online batch algorithm on the straight-planar, sinusoidal-planar, straight-sinusoidal-planar, sinusoidal-hill, and straight-sinusoidal-hill trajectory. The algorithmic and simulation parameters used in the experiments are reported in Tab. 4.3 and Tab. 4.4.

We display in Fig. 4.33 the selected informative batches of data for the calibration. As compared to the application in Sec. 4.2, we observe that the online algorithm requires more data to converge, even though we have fixed the

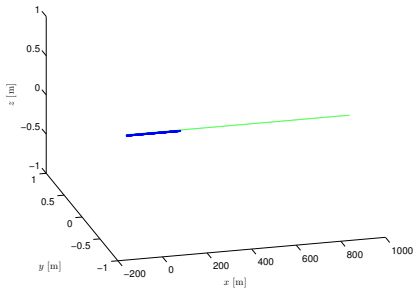
ΔN	δ	u_s	u_{ψ}	$\epsilon_{\boldsymbol{\theta}}$	t_{max}	ε
200	0.5	10^{-16}	10^{-16}	10^{-5}	20	10^{-4}

Table 4.3: Algorithmic parameters for the experiments. We refer the reader to Sec. 3.4 for the signification of these parameters.

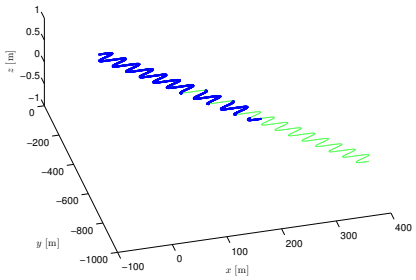
Parameter	Value
N	10000
T	0.1 [s]
$\sigma^2_{w_{rl/rr/fl/fr/dmi/v}}, \mathbf{\Sigma}^r$	10^{-6}
$\sigma^2_{l/n/\omega_x/\omega_y}$	10^{-3}
s^o	4
s^k	5 [s^{-1}]
e_r, e_f	0.75 [m]
L	2.7 [m]
$\mathbf{a}$	$a_0 = 0, a_1 = 572.9578, a_2 = 0, a_3 = 0$
$k_{rl}, k_{rr}, k_{fl}, k_{fr}$	360
k_{dmi}	1
${}^v\mathbf{T}_r$	${}^v\mathbf{R}_r = \mathbf{I}$, ${}^v r_1^{vr} = 0$ [m], ${}^v r_2^{vr} = 0$ [m], ${}^v r_3^{vr} = 0.8$ [m]

Table 4.4: Simulation parameters for the experiments. More details in Sec. 4.3.1.

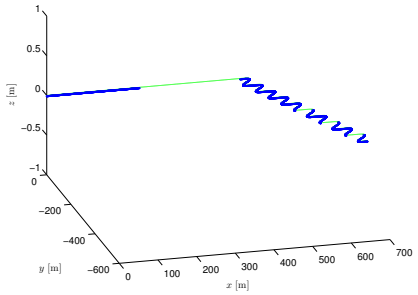
information threshold δ to a slightly higher value. We suspect that the higher dimensionality of the calibration parameters might play a role in this behavior. Apart from that, these plots are particularly instructive and confirm our a priori analysis. For Fig. 4.33a, Fig. 4.33b, and Fig. 4.33c, the batch selection is similar to the LRF example (*cf.* Fig. 4.17). We witness in Fig. 4.33d and Fig. 4.33e that our algorithm picks data in regions where the car's pitch angle is changing. This is indeed the only situation where the third component of ${}^{v}\mathbf{r}^{vr}$ becomes observable.



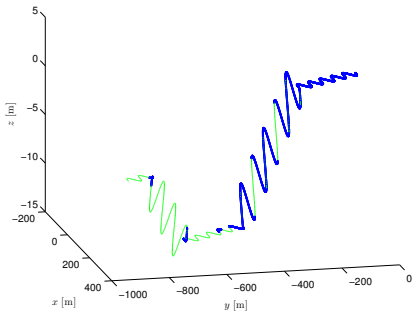
(a) Straight-planar trajectory.



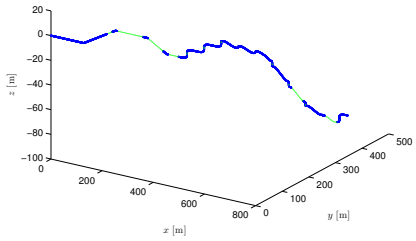
(b) Sinusoidal-planar trajectory.



(c) Straight-sinusoidal-planar trajectory.



(d) Sinusoidal-hill trajectory.

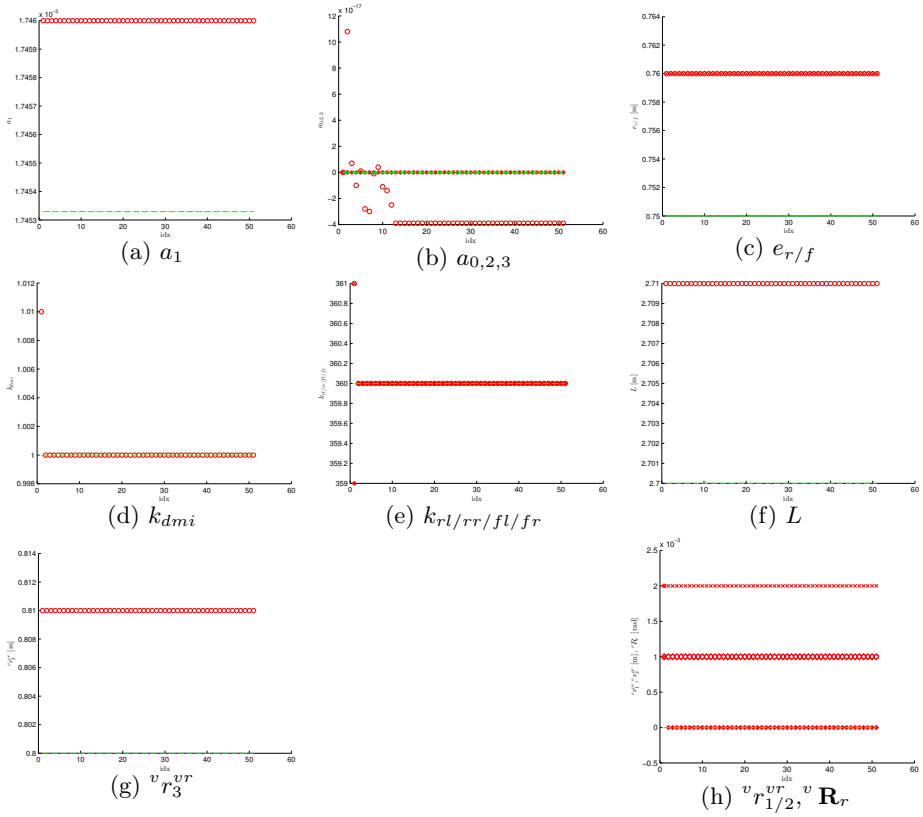


(e) Straight-sinusoidal-hill trajectory.

Figure 4.33: Qualitative results for the estimation of vehicles's poses for simulated experiments. The ground truth trajectory ${}^{m}\mathbf{T}_{v_{1:N}}$ is shown in green and trajectory estimate ${}^{m}\widehat{\mathbf{T}}_{v_{1:N}}$ in blue.

For the sake of clarity, we only show the evolution of the calibration parameters for three representative cases in Fig. 4.34, Fig. 4.35, and Fig. 4.36. As desired, we observe that the parameters remain at their initial guess until they become observable. The scale factors are well estimated in all the configura-

tions; the axle tracks, wheel base, and steering coefficients require turns, and ${}^{v}r_3^{vr}$ changes in pitch angle. We see in Fig. 4.35f, Fig. 4.35h and Fig. 4.36f, Fig. 4.36h an interesting link between the offset in L and ${}^{v}r_1^{vr}$ due to their coaxial arrangement. Furthermore, the variability of ${}^{v}r_3^{vr}$ in Fig. 4.36g originates from the fact that the parameter is weakly observable, as was already reflected in the singular values spectrum in Fig. 4.32.



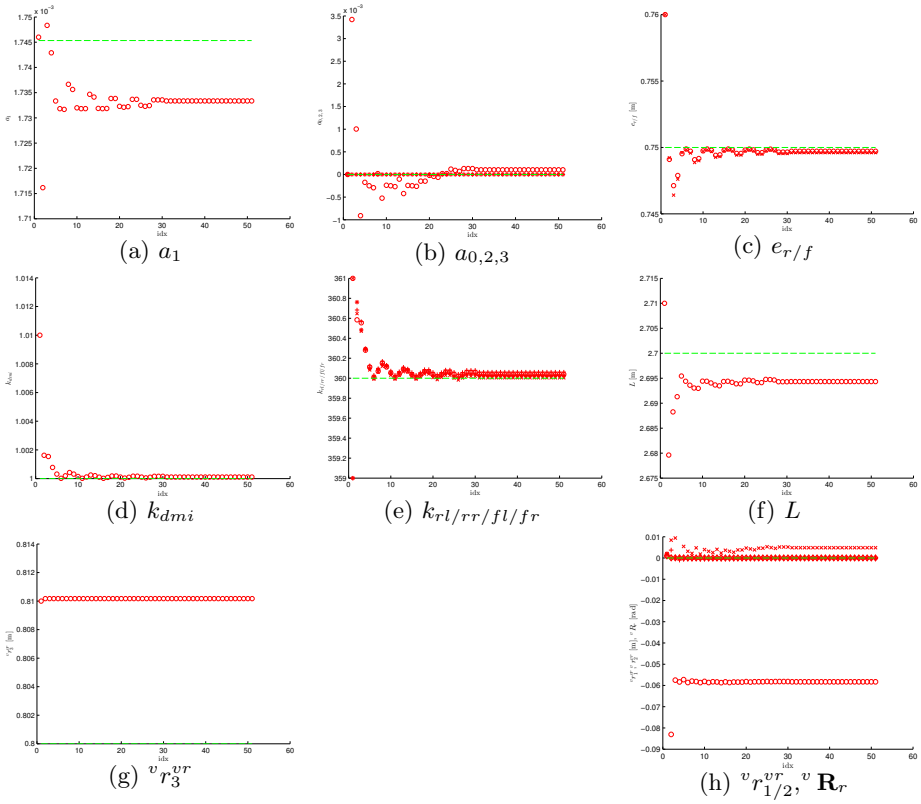
(a) a_1 (b) $a_{0,2,3}$ (c) $e_{r/f}$

(d) k_{dmi} (e) $k_{rl/rr/fl/fr}$ (f) L

(g) ${}^{v}r_3^{vr}$ (h) ${}^{v}r_{1/2}^{vr}$, ${}^{v}\mathbf{R}_r$

Figure 4.34: Evolution of the calibration parameters $\widehat{\boldsymbol{\theta}}$ as a function of the incoming data batches for the straight-planar trajectory. The ground truth is shown in green. In this configuration, only the wheel scale factors (Fig. 4.34d and Fig. 4.34e) and the first polynomial coefficient of the steering wheel transmission (red circles in Fig. 4.34b) are observable. The other parameters remain at their initial guess.

To conclude these experiments, we report in Fig. 4.37 the evolution of the information gain as new batches are added to the online estimator. As compared to Sec. 4.2, we have not set the information gain to infinity when the observable subspace dimension increases. Nevertheless, the positions of the peaks correspond to batches of high information. As reflected in the singular values spectrum, pitch angle changes are not associated with high information gain, in comparison to yaw angle variations.

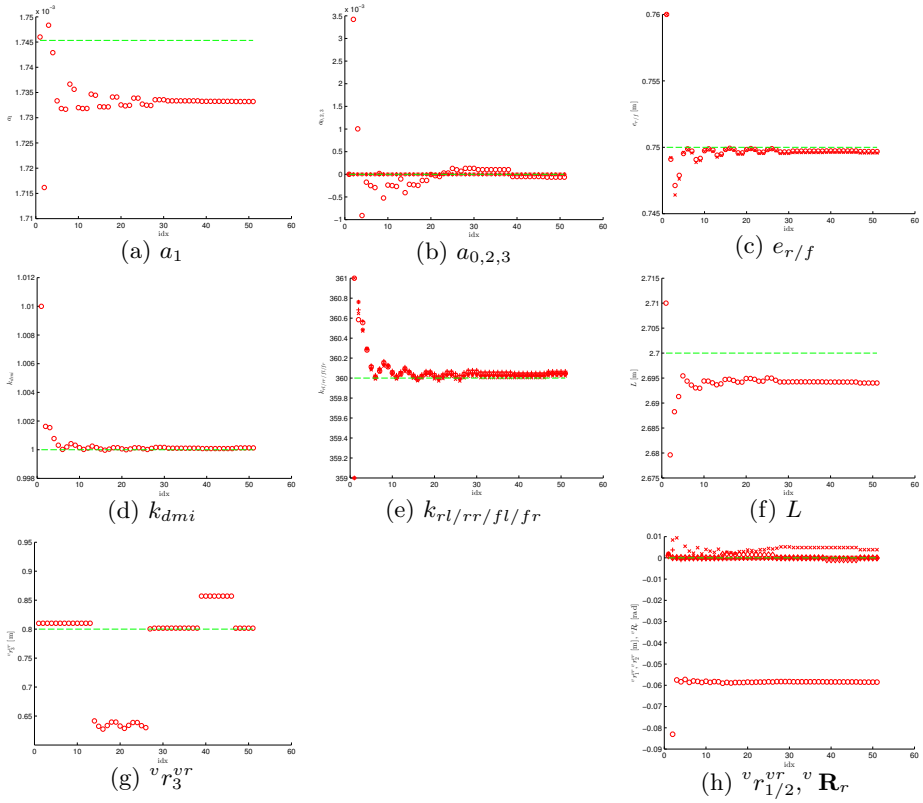


(a) a_1 (b) $a_{0,2,3}$ (c) $e_{r/f}$

(d) k_{dmi} (e) $k_{rl/rr/fl/fr}$ (f) L

(g) ${}^{v}r_3^{vr}$ (h) ${}^{v}r_{1/2}^{vr}$, ${}^{v}\mathbf{R}_r$

Figure 4.35: Evolution of the calibration parameters $\widehat{\boldsymbol{\theta}}$ as a function of the incoming data batches for the sinusoidal-planar trajectory. The ground truth is shown in green. As compared to Fig. 4.34, we observe here that the remaining polynomial coefficients of the steering wheel transmission (Fig. 4.35a and Fig. 4.35b), the wheel track (Fig. 4.35f), the wheel axles (Fig. 4.35c), the first two components of ${}^{v}\mathbf{r}^{vr}$ (circles and crosses in Fig. 4.35h), and the rotation ${}^{v}\mathbf{R}_r$ (plus, stars, and diamonds in Fig. 4.35h) are observable. From Fig. 4.35f and Fig. 4.35h, we notice a coupling between the wheel track and the first component of ${}^{v}\mathbf{r}^{vr}$.

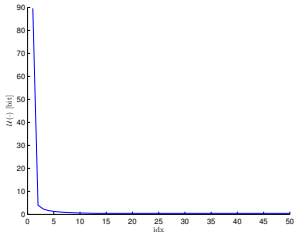


(a) a_1 (b) $a_{0,2,3}$ (c) $e_{r/f}$

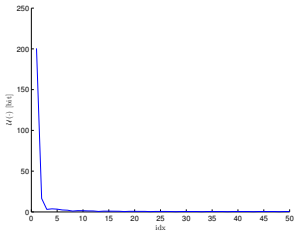
(d) k_{dmi} (e) $k_{rl/rr/fl/fr}$ (f) L

(g) ${}^{v}r_3^{vr}$ (h) ${}^{v}r_{1/2}^{vr}$, ${}^{v}\mathbf{R}_r$

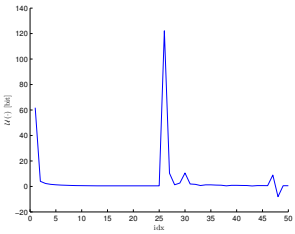
Figure 4.36: Evolution of the calibration parameters $\widehat{\boldsymbol{\theta}}$ as a function of the incoming data batches for the sinusoidal-hill trajectory. The ground truth is shown in green. As compared to Fig. 4.35, the remaining calibration parameter ${}^{v}r_3^{vr}$ (Fig. 4.36g), corresponding to the third component of ${}^{v}\mathbf{r}^{vr}$, becomes observable. The variations occurring in the estimates show that this parameter is weakly observable.



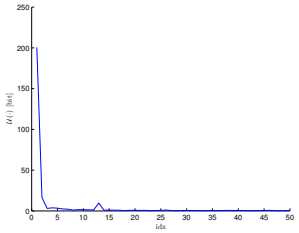
(a) Straight-planar trajectory.



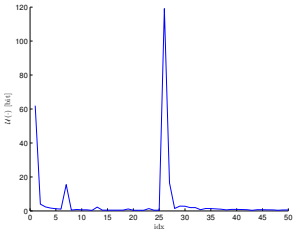
(b) Sinusoidal-planar trajectory.



(c) Straight-sinusoidal-planar trajectory.



(d) Sinusoidal-hill trajectory.



(e) Straight-sinusoidal-hill trajectory.

Figure 4.37: Evolution of the information gain with respect to incoming batches for the different configurations in the car example.

4.3.4 Real-World Experiments

After these simulations, we shall investigate whether the results agree with real-world experiments and whether our model is a sufficient approximation of the physical system. As in Sec. 4.2.4, we lack ground truth data for the calibration parameters. This is a general problem when it comes to evaluating calibration algorithms on real-world data. However, we might compare the results across different datasets and perform the calibration with alternative methods. Keeping in mind the kind of parameters we want to calibrate, a relevant measure is the quality of the integrated odometry trajectory, before and after calibration. Finally, we might evaluate the predictive accuracy of the calibrated parameters on unseen data.

Concerning the initial guesses for the calibration parameters, we have inspected the Toyota Prius datasheet for the rear and front axle tracks, and for the wheelbase. For the scale factors on the WSS and DMI, along with the coefficients on the SWS, we have inferred their value from the data. For the transformation between the Applanix and vehicle frame, we have used the parameters resulting from the initial calibration performed by Applanix's engineers. All in all, the initial calibration parameters are summarized in Tab. 4.5. As earlier, the numerous variances entering the calibration algorithm have been estimated from a training dataset. Lastly, the algorithmic parameters follow Tab. 4.3.

4.3.5 Parking Lot Dataset

Before starting, it is worth inspecting the kind of data we are dealing with in this problem. In Fig. 4.38, we show all the involved sensory data for the first dataset under consideration. This dataset of approximatively 2 minutes was recorded in a parking lot and contains various changes of heading, but mostly planar

Parameter	Value
e_r	0.74041 [m]
e_f	0.75311 [m]
L	2.70002 [m]
$\mathbf{a}$	$a_0 = 0, a_1 = 572.9578, a_2 = 0, a_3 = 0$
$k_{rl}, k_{rr}, k_{fl}, k_{fr}$	360
k_{dmi}	1
${}^v\mathbf{r}^{vr}$	${}^v r_1^{vr} = 0$ [m], ${}^v r_2^{vr} = 0$ [m], ${}^v r_3^{vr} = 0.785$ [m]
${}^v\mathbf{R}_r$	${}^v R_{r_1} = 0$ [rad], ${}^v R_{r_2} = 0$ [rad], ${}^v R_{r_3} = \pi$ [rad]

Table 4.5: Initial guesses for the real-world experiments in the car application. More details on the parameters in Sec. 4.3.1.

motion. Except a vertical jump at the start of the trajectory, the Applanix pose data is relatively smooth. This discontinuity is due to the internal filter of the sensor which uses the DMI sensor to stabilize the pose when the vehicle is static and transits to GPS during motion. Apart from being quantized, the CAN WSSs only start emitting nonzero data at a velocity of approximatively 1 [m/s] at a frequency of 60 [Hz]. The CAN SWS sends out measurements at 30 [Hz]. While we rely on the acquisition time for CAN data (+/− 0.01 [s]), we have implemented a filter based on GPS time for the Applanix data (+/− 0.0005 [s]).

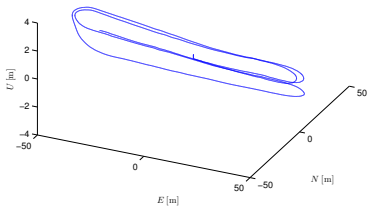
We display in Fig. 4.39 the estimated path for the first dataset, along with the evolution of the information gain. As we witnessed in Sec. 4.3.3, our online algorithm requires more data samples to converge. For this short dataset, it retained 5 out of the 6 available batches.

The final estimate $\widehat{\boldsymbol{\theta}}$ is summarized in Tab. 4.6. In this table, we also compare the results with a full batch method. We retrieve here the same kind of connection between the wheelbase L and ${}^v r_1^{vr}$. To complete this presentation, we also display the integrated odometry trajectory before and after calibration in Fig. 4.40. Finally, we have obtained similar results for three other datasets of the same kind (not displayed here), hence validating our estimates.

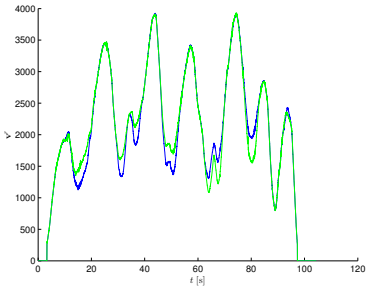
4.3.6 City Tour Dataset

To conclude our experimental evaluation for the car application, we have chosen the challenging dataset depicted in Fig. 4.41. This dataset contains approximatively 20 minutes of car driving in a city with various changes of orientation. While our algorithm remains tractable, we notice here that a standard batch method reaches the computational and storage limit of a high-end machine.

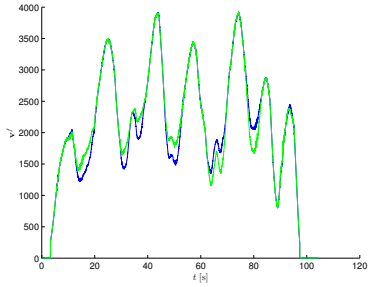
Following the same methodology as above, we first outline in Fig. 4.42 the estimated path and the evolution of the information gain. As opposed to the parking lot dataset, we observe here that only half of the data batches were necessary for the algorithm to converge.



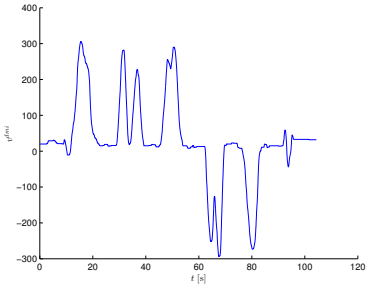
(a) Applanix path.



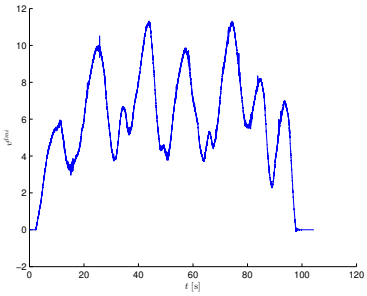
(b) CAN rear WSS left (blue) and right (green).



(c) CAN front WSS left (blue) and right (green).

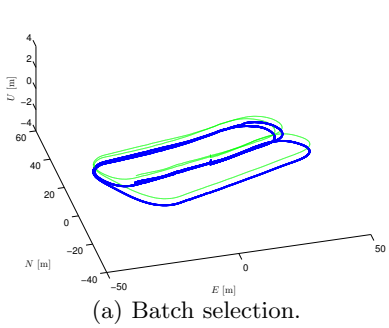


(d) CAN SWS.

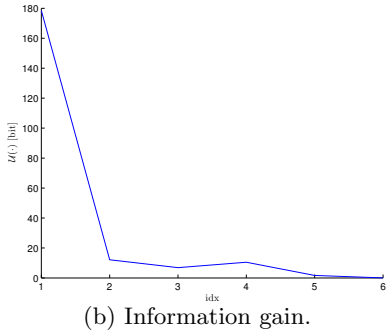


(e) Applanix DMI.

Figure 4.38: Sensory data for the parking lot dataset.



(a) Batch selection.



(b) Information gain.

Figure 4.39: Qualitative results for the dataset in Fig. 4.38. Fig. 4.39a shows the estimated path ${}^{m}\widehat{\mathbf{T}}_{v_{1:N}}$ (blue) and the Applanix path (green). Fig. 4.39b visualizes the evolution of the information gain as a function of the incoming batches.

Parameter	Online Batch	Offline Batch
e_r	0.74394	0.74354
e_f	0.73993	0.74023
L	2.62867	2.60404
$\mathbf{a}$	$-0.02108, 0.00123, 0, 0$	$-0.02099, 0.00123, 0, 0$
$k_{rl/rr}$	$354.1539, 353.75263$	$354.208396, 353.82819$
$k_{fl/fr}$	$355.08913, 355.10648$	$355.30116, 355.27483$
k_{dmi}	1.01516	1.01541
${}^v\mathbf{r}^{vr}$	$-0.22325, 0.02026, 0.64082$	$-0.23027, 0.02069, 0.70919$
${}^v\mathbf{R}_r$	$0.00446, -0.02839, 3.13981$	$0.004647, -0.028726, 3.13916$

Table 4.6: Quantitative results for the parking lot dataset with our method (left) and a standard batch method (right).

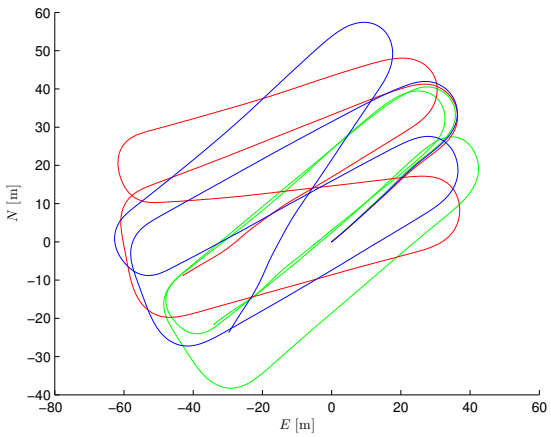
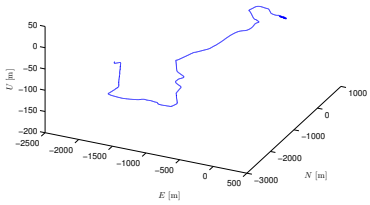
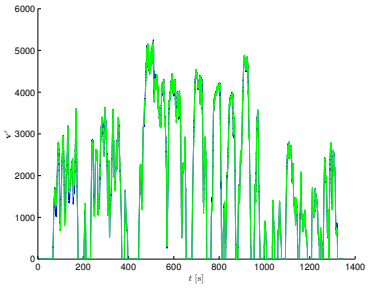


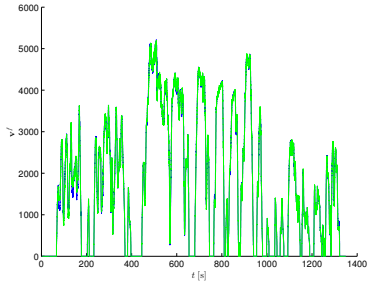
Figure 4.40: Odometry integration for the parking lot dataset. The Applanix trajectory is shown in green, the integrated odometry path before calibration in red, and after in blue. For the straight portions of the path, the red (before calibration) and blue (after calibration) trajectories are comparable and match the Applanix trajectory. For each turn, the red path overshoots the steering angle significantly more than the blue path.



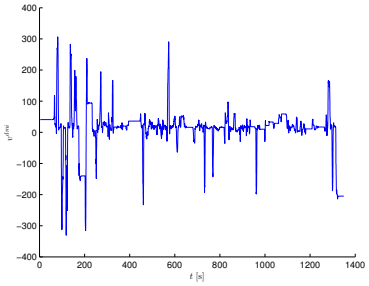
(a) Applanix path.



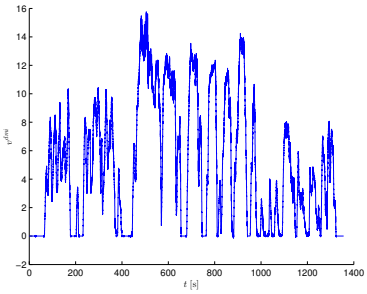
(b) CAN rear WSS left (blue) and right (green).



(c) CAN front WSS left (blue) and right (green).

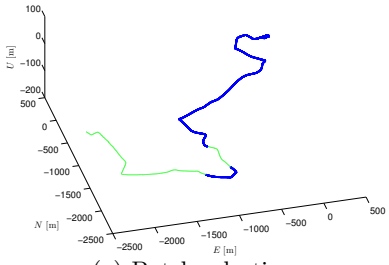


(d) CAN SWS.

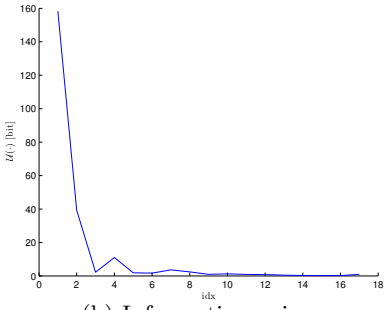


(e) Applanix DMI.

Figure 4.41: Sensory data for the city tour dataset.



(a) Batch selection.



(b) Information gain.

Figure 4.42: Qualitative results for the dataset in Fig. 4.41. Fig. 4.42a shows the estimated path ${}^{m}\widehat{\mathbf{T}}_{v_{1:N}}$ (blue) and the Applanix path (green). Fig. 4.42b visualizes the evolution of the information gain as a function of the incoming batches.

As earlier, we have gathered the final estimate $\widehat{\boldsymbol{\theta}}$ in Tab. 4.7, where we also compare our algorithm with a full batch method. It is pleasing to notice that these estimates are in line with the parking lot dataset, although the two datasets were recorded 4 months apart. Finally, we have plotted the integrated odometry trajectories in Fig. 4.43.

Parameter	Online Batch	Offline Batch
e_r	0.74401	0.74296
e_f	0.75152	0.74542
L	2.69660	2.65956
$\mathbf{a}$	$0.02264, 0.0013, 0, 0$	$-0.02227, 0.00128, 0, 0$
$k_{rl/rr}$	$354.71496, 353.83646$	$354.58787, 353.81908$
$k_{fl/fr}$	$354.20178, 354.03252$	$354.41196, 354.28826$
k_{dmi}	1.01553	1.01525
${}^v\mathbf{r}^{vr}$	$-0.17807, 0.02355, 0.643145$	$-0.20107, 0.00588, 0.71431$
${}^v\mathbf{R}_r$	$0.00325, -0.0163, 3.14149$	$0.00374, -0.016804, 3.1415$

Table 4.7: Quantitative results for the city tour dataset with our method (left) and a standard batch method (right).

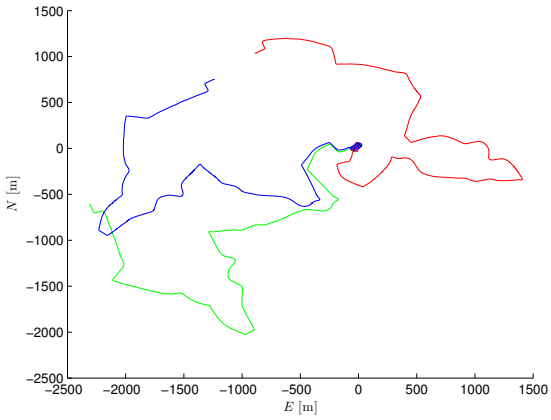


Figure 4.43: Odometry integration for the city tour dataset. The Applanix trajectory is shown in green, the integrated odometry path before calibration in red, and after in blue. We observe here a significant improvement between the uncalibrated (red path) and calibrated (blue path) parameters.

4.4 Camera Calibration

Our last application concerns the calibration of a digital camera. Being comparably inexpensive, cameras have become an affordable substitute to laser-based technologies in robotic or mapping applications. In this context, digital images produced by a camera are only relevant if they can be related to the real physical objects they picture, *e.g.*, for robotic manipulators. This entails the accurate determination of the intrinsic and extrinsic parameters of the camera. While these latter consist in the relative pose of the camera with respect to a world frame, the former specify how the world points get projected onto the imaging sensor. If the physical setup of the camera is fixed, the camera intrinsics can be reused in other scenes, *e.g.*, for canceling the effect of cheap lenses.

Camera calibration has been an intensive subject of research since the advent of digital cameras in the seventies. In this timespan, various techniques have been proposed, with diverse accuracy and complexity of the calibration setup. For our application, we build on the approach pioneered by Zhang [Zha99]. This method requires a planar calibration pattern with known geometry, *e.g.*, a checkerboard (Fig. 4.44b), that is observed from the camera (Fig. 4.44a) under different poses. Extracted features in the images, using for instance a corner detector [HS88, RSS08], can be associated at no cost to points in the three-dimensional Euclidean space. The forward model can therefore be understood as the function that maps points in the calibration pattern to images features and the calibration process as the search for the inverse model.

(a) MATRIX VISION camera.

(b) Checkerboard detection.

Figure 4.44: Illustration of the sensor (Fig. 4.44a) and the checkerboard (Fig. 4.44b) for the camera calibration problem. The corners of the checkerboard are detected (colored points) for each view of the calibration pattern. In this scenario, the calibration problem consists in determining the intrinsic parameters of the camera, *i.e.*, the lens characteristics.

In the rest of this section, we shall first state how the camera calibration problem fits into our generic online calibration algorithm. After a formal introduction, we will evaluate the potential of our approach to handle this task by means of experiments and comparisons to state-of-the-art methods. Camera calibration being a well-understood problem, we directly focus on real-world data and mention observability issues whenever they occur.

4.4.1 Problem Formulation

For each view k, we attach a three-dimensional coordinate system $\mathcal{F}_{O_k}$ to the planar calibration pattern, with origin o_k at one corner of the pattern, such that its first and second axes are aligned with the pattern's contours. Each point ${}^{o_k}\mathbf{p}_i = ({}^{o_k}p_1, {}^{o_k}p_2, 0)^T$ in the calibration pattern is associated with a point ${}^{p}\mathbf{p}_i = ({}^{p}c, {}^{p}r)^T$ expressed in *pixel coordinates* with respect to a row-column pixel frame $\mathcal{F}_P$, whose origin p is at the top-left corner of the virtual image plane, first axis pointing to the right and second axis to the bottom of the image plane.

Observation Model

In this application, we shall first consider the standard *pinhole projection model*, which consists in a composition of affine transformation, *projections*, and *digitization* or *quantization*. To account for the effects of the optical lens, *i.e.*, *radial* and *tangential distortions*, a distortion stage is added. Within this projection model, we shall define a frame $\mathcal{F}_C$ with origin c located at the *perspective center* of the camera, first and second axes aligned with $\mathcal{F}_P$, and third axis in the direction of the virtual image plane. Finally, we establish a frame $\mathcal{F}_I$ with origin located at the *principal point* or *image center* of the image, *i.e.*, the intersection point of the third axis of $\mathcal{F}_C$ with the image plane. Hence, the forward model for ${}^{p}\mathbf{p}_i$ can be expressed as

$$ {}^{p}\mathbf{p}_i \quad = \quad h_p({}^{o_k}\mathbf{p}_i, {}^{c}\mathbf{T}_{o_k}, k_1, k_2, p_1, p_2, f, s_x, s_y, c_o, r_o), \tag{4.54} $$

where ${}^{o_k}\mathbf{p}_i$ is known, ${}^{c}\mathbf{T}_{o_k}$ transforms points from $\mathcal{F}_{O_k}$ to $\mathcal{F}_C$, $\mathbf{d} = (k_1, k_2, p_1, p_2)^T$ are the radial and tangential distortion parameters, f is the focal length, $(s_x, s_y)^T$ are the spatial quantization factors, ${}^{p}\mathbf{c} = (c_o, r_o)^T$ are the pixel coordinates of the principal point. The focal length and the quantization factors shall be combined, due to the arbitrary scale factor, as $\mathbf{f} = (f_x, f_y) = (fs_x, fs_y)^T$. While $\mathcal{I} = \{\mathbf{d}, \mathbf{f}, {}^{p}\mathbf{c}\}$ will be referred to as intrinsic parameters, $\mathcal{E}_k = \{{}^{c}\mathbf{T}_{o_k}\}$ contains the extrinsic parameters. We refer the reader to [HZ03] for a more in-depth treatment of this projection model. The setup of the coordinate frames for the pinhole projection model is summarized in Fig. 4.45.

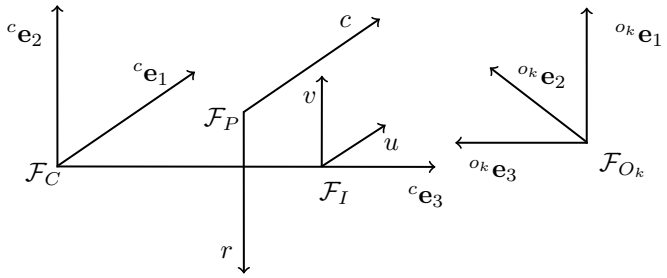


Figure 4.45: Setup of the coordinate frames for the pinhole projection model.

The pinhole projection model is suboptimal for *wide-angle lenses*, *fish-eye lenses*, or *catadioptric sensors*. In our experiments, we shall therefore also evaluate the generalized projection model presented in [MR07]. The point ${}^{o_k}\mathbf{p}_i$

is first transformed into a mirror frame $\mathcal{F}_M$, then projected onto a unit sphere centered at m and expressed in a reference frame shifted by ξ along the third axis of $\mathcal{F}_M$, and finally projected onto $\mathcal{F}_P$ using $\mathcal{I}$ defined above. The value of ξ depends on the shape of the lens or the mirror; if $\xi = 0$, we retrieve the standard perspective projection model. Consequently, the forward model for the generalized projection model can be expressed as

$$ {}^{p}\mathbf{p}_i \;=\; h_g({}^{o_k}\mathbf{p}_i,\, {}^{m}\mathbf{T}_{o_k}, \mathcal{I}, \xi), \tag{4.55} $$

where ${}^{m}\mathbf{T}_{o_k}$ is now the transformation from $\mathcal{F}_{O_k}$ to $\mathcal{F}_M$. The different coordinate frames involved in the generalized projection model are shown in Fig. 4.46.

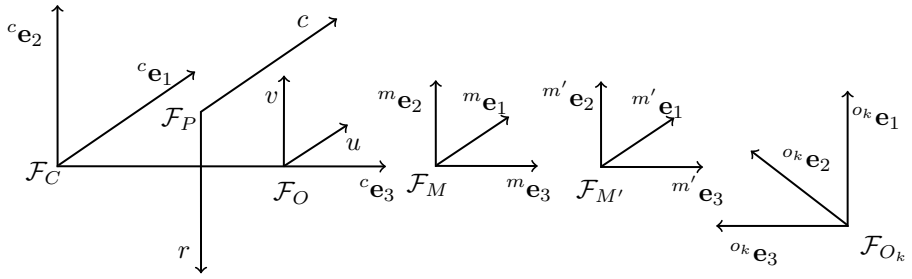


Figure 4.46: Setup of the coordinate frames for the generalized projection model.

Estimation Model

With respect to our calibration algorithm, the calibration variable is $\boldsymbol{\Theta}_p = \{\mathcal{I}\}$ for the pinhole model and $\boldsymbol{\Theta}_g = \{\mathcal{I}, \xi\}$ for the generalized model. Given a collection of N views of the calibration pattern, the nuisance variable becomes $\boldsymbol{\Psi}_p = \{{}^{c}\mathbf{T}_{o_1}, \ldots, {}^{c}\mathbf{T}_{o_N}\}$ or $\boldsymbol{\Psi}_g = \{{}^{m}\mathbf{T}_{o_1}, \ldots, {}^{m}\mathbf{T}_{o_N}\}$. For a given view k, we define the error terms for the point ${}^{p}\mathbf{p}_i$ observed in the image and associated with ${}^{o_k}\mathbf{p}_i$ as

$$ \begin{aligned} \mathbf{e}^{p}_{k_i} &= {}^{p}\mathbf{p}_i - h_p({}^{o_k}\mathbf{p}_i,\, {}^{c}\mathbf{T}_{o_k}, \mathcal{I}), \\ \mathbf{e}^{g}_{k_i} &= {}^{p}\mathbf{p}_i - h_g({}^{o_k}\mathbf{p}_i,\, {}^{m}\mathbf{T}_{o_k}, \mathcal{I}, \xi), \end{aligned} \tag{4.56} $$

such that

$$ \begin{aligned} \mathbf{e}^{p}_{k_i} &\sim \mathcal{N}(\mathbf{0}, \sigma_p^2 \mathbf{I}), \\ \mathbf{e}^{g}_{k_i} &\sim \mathcal{N}(\mathbf{0}, \sigma_g^2 \mathbf{I}), \end{aligned} \tag{4.57} $$

and $\sigma^2_{p/g}$ are deterministic variance parameters that are a priori estimated. As for Sec. 4.3, we skip the derivations of the Jacobian matrices and the representation of the joint probability density function. Initial guesses for the latent variables can be obtained by the methods presented in the aforementioned references.

4.4.2 Experiments

Since we have demonstrated the different insights of our algorithm to a large extent in Sec. 4.2 and Sec. 4.3, we shall directly focus on real-world data for this camera calibration application. We have recorded datasets of checkerboard images with a global shutter camera from MATRIX VISION, equipped with a normal or wide-angle lens. For comparison purpose, we have used two baseline algorithms, the ROS-OpenCV calibration toolbox and the CamOdoCal software [HLP13]. It is worth noting that the ROS package is endowed with a heuristic for selecting a minimal set of images that are necessary for proper calibration of the sensor. Their scheme consists in picking images with the following characteristics:

- Lateral motion of the checkerboard spanning the entire image plane.
- Vertical motion of the checkerboard spanning the entire image plane.
- Tilting of the checkerboard at different orientations.
- Checkerboard at various scales, from far to near, covering the entire image plane.

We have observed through the experiments that our online batch algorithm indeed selects the images according to this principle. Consequently, the ROS algorithm, based on purely geometric arguments, is supported by an information-theoretic justification. In addition to the pinhole projection model, the CamOdoCal software implements the generalized projection model [MR07]. While the degenerate cases for the pinhole projection model are outlined in [LRAAD10], the generalized projection model has singularities for points projected close to the image center, as we shall see in the experiments.

For the first experiment, we have mounted a normal lens on the camera and evaluated the calibration algorithms with the pinhole projection model. The dataset contains approximatively 2000 images covering all the required positions. The calibration results are summarized in Tab. 4.8 for the pinhole projection model. We observe that our algorithm is comparable with CamOdoCal in terms of focal lengths and image centers. The ROS algorithm yields slightly different results.

In a second experiment, we have used the same dataset, but calibrated against the generalized projection model. In that case, only our algorithm and CamOdoCal provide results, summarized in Tab. 4.9. For this estimation problem, our algorithm has detected an unobservable direction spanning ξ, f_x, and f_y, conformed to our intuition. While we obtain similar results across all the recorded datasets, CamOdoCal becomes instable in this degenerate case, with f_x or f_y reaching up to 10000 in some cases. Under the hood, CamOdoCal uses Levenberg-Marquardt for the nonlinear optimization. By adding a full-rank diagonal matrix weighted by a dampening parameter to the Jacobian matrix, the Levenberg-Marquardt algorithm can mimic a regularization tool. However, the behavior becomes instable when approaching a minimum and the Jacobian matrix is rank-deficient. As explained in Sec. 2.1.6, the primary goal of the dampening parameter is to balance the update steps between gradient

Parameter	Online Batch	CamOdoCal	ROS
f_x	1229.03608	1229.06481	1123.91784
f_y	1224.17118	1224.19565	1166.96765
c_o	650.59084	650.57302	651.68636
r_o	501.24995	501.16402	499.58002
k_1	−0.31474	−0.31489	−0.31219
k_2	0.12708	0.12728	0.12077
p_1	0.00054	0.00055	0.00094
p_2	−0.00006	−0.00008	−0.00033

Table 4.8: Calibration parameters for the pinhole projection model with a normal lens.

descent and Gauss-Newton. Close to a minimum, the dampening parameter should vanish to favor Gauss-Newton. In a rank-deficient case, the parameter will hence have an oscillatory behavior.

Parameter	Online Batch	CamOdoCal
ξ	0.88948	0.98409
f_x	2321.95213	2443.98029
f_y	2312.84065	2434.76122
c_o	650.63825	648.36969
r_o	501.71634	502.56365
k_1	−0.31779	−0.29689
k_2	0.16479	0.18315
p_1	0.00093	0.96317
p_2	−0.00006	−0.00005

Table 4.9: Calibration parameters for the generalized projection model with a normal lens.

5 Conclusion

What we know is not much. What we do not know is immense.

Pierre-Simon Laplace (1749-1827)

The last chapter of this thesis shall be dedicated to recapitulating the major contributions and insights following from the previous chapters. Starting from a summary of the most important aspects of our work, we shall open up a discussion about its limitations and the potential research directions to address them. Finally, we want to acknowledge the persons who have been involved directly or indirectly in the elaboration of this document.

5.1 Summary

This thesis has provided various insights into the topic of sensor calibration for robotic systems. To fulfill this task, we have investigated the development of an automated method that deals with degenerate configurations occurring during calibration runs. By means of matrix analysis, our algorithm identifies unobservable directions in the calibration parameter space and locks them to their initial guess until they become fully observable. To leverage the otherwise intractable computational and space complexity of nonlinear optimization methods, we have proposed an information-theoretic scheme that evaluates the utility of data sample batches for the calibration. To elaborate our algorithm, we have conciliated numerous research fields, notably Bayesian and frequentist statistics, linear algebra and numerical analysis, information theory, control theory and robotics. We have anchored our method with theoretical aspects borrowed from all these fields, which are usually hermetic to each other, despite the fundamentally similar problems they are tackling.

The first contribution of this work concerns an online observability measure of the parameter space. Whilst the majority of state-of-the-art approaches are either not aware of singularities or perform observability analysis in an offline phase, we have presented a method based on rank-revealing QR and singular value decompositions of the Fisher information matrix that separates the calibration parameters into observable and unobservable parts. Splitting up the parameter space, our algorithm only updates the observable dimensions, leaving the rest intact. Along these lines, we have developed the concept of numerical observability, the counterpart of numerical rank in linear algebra, as opposed to the purely algebraic view in control theory. Through our simulated experiments, we have illustrated the difficulty in assessing observability in presence of noisy sensor data and shown the robustness of our algorithm to properly execute this task in comparison to other methods.

In a second effort, we have validated the theoretical intuitions on the behavior of the entropy difference between prior and posterior distributions when new data batches enter the estimator through simulated and real-world experiments. Linked to the observability, this measure perfectly fulfills our expectations for

the selection of informative data and is hence a sound complement to nonlinear least-squares methods.

Finally, we have showcased three prospective applications and demonstrated the potential of our algorithm to solve real-world problems. The first application concerned the extrinsic calibration of a laser rangefinder with respect to a differential-drive robot, the second application intended to estimate the odometry parameters of a consumer car based on native CAN bus data, and the last application targeted the classical camera intrinsics calibration. For all these applications, we have didactically presented the setup and validated our algorithm through experiments.

To conclude, we have successfully addressed several relevant aspects of the sensor calibration problem. We notably believe in the potential of our approach for the unsupervised calibration of consumer-level sensors and we hope it can open up further lines of research in that direction. With the advent of close-to-market autonomous cars, sensor calibration will definitely remain a critical task to ensure the safety of the users. For these applications, the algorithms need to be robust on the one hand and on the other hand to require little supervision.

In a more visual representation, the contributions of this thesis can be summarized as follows:

- Data-driven observability analysis for calibration problems based on matrix analysis
- Sequential batch algorithm based on a entropy measure for long-term operation
- Evaluation on three challenging applications

5.2 Limitations and Improvements

The presented algorithm for the online self-calibration of robotic sensors has several limitations and therefore offers vast opportunities for further research directions that we shall discuss in the rest of this section.

5.2.1 Discussion

The key limitation in the current implementation lies in the assumption that the calibration parameters remain constant during the calibration run. For the proper deployment of our method into consumer-level sensors, we shall be able to cope with slowly to abruptly varying parameters. As an example, we can consider an autonomous vehicle with cameras embedded into wing mirrors. Whenever the user opens or closes the doors, the relative poses of the cameras might change due to the mechanical shock. It is nevertheless possible to detect these cases by monitoring sudden variations in the prediction error of the estimator and restarting the calibration algorithm from scratch. A probably smarter approach would be to add a forgetting factor to our algorithm, in a similar vein than what was presented for the adaptive controllers or the online sparsification method in Sec. 1.2. Using this strategy, the forgetting factor would give exponentially less weight to older data batches, discarding them

eventually, to favor fresh information. The covariance matrix of the estimates should be scaled accordingly to avoid a too strong convergence of our online algorithm, leading to discounting any novel data.

A current limitation of our algorithm is the dependence on a reasonable initial guess for the calibration parameters. To reach the fully unsupervised goal we have in mind, the calibration algorithm should be able to bootstrap itself from scratch. Along the same lines, our implementation uses Gauss-Newton update steps in the nonlinear optimization, which is hence only guaranteed to converge if the initial guess is close enough to a local minimum. To withdraw this assumption, we suggest a trust-region method such as Powell's dog-leg algorithm [Pow70]. This class of methods uses a combination of gradient descent steps for global convergence and Gauss-Newton steps for local convergence. Our first experience with this method was disappointing since the gradient descent steps were updating the parameters along the unobservable directions. However, we have sketched a regularized version of the algorithm that we shall rigorously validate before releasing it.

The determination of the thresholds for the numerical rank detection still remains debatable in the current implementation of our algorithm. As we have seen in Chap. 4, since these thresholds are clearly dependent on the system noise, there must be a way to compute them analytically. From our experiments, we have nevertheless witnessed that these thresholds only need to be calculated once per sensor setup, which remains a manageable task for an expert before the deployment of the algorithm in the field. Skimming through the literature, we have observed that even experts [Dav11] in numerical analysis employ a heuristic that "has worked well in practice". It is important to recall here that an incorrect numerical rank detection can lead to utterly wrong estimates as was demonstrated in Sec. 4.2. In that regard, we suggest a conservative approach, such that these thresholds may underestimate the numerical rank rather than overestimating it.

For some problems, the calibration parameters may need to be constrained in a specific range, *e.g.*, masses should be strictly positive in a rigid body dynamics scenario. Our algorithm has no internal mechanism to enforce these constraints in the current implementation and we might therefore end up with incoherent physical quantities. In order to cope with this limitation, the nonlinear optimization framework should be entirely revised or a sufficiently concentrated prior should be assigned to these parameters.

As we have chosen an entropy-based measure to select informative measurements in our online algorithm, it might be worth investigating the feasibility of the promising approach presented in [NTP11], which inserts or removes data in a dictionary based on a linear independence test. In contrast to our method, the linear independence test applies prior to estimation. In our setup, this test should be executed on the columns of the Jacobian matrix. The method should nevertheless be adapted as the columns might appear linearly independent due to the chosen linearization point.

We shall conclude this discussion with some limitations introduced by the normality assumption on the sensor data distributions. In case this assumption is violated, we have presented in Sec. 2.1.5 several popular robust estimation techniques that cope with outliers. For our algorithm, we have adopted the M-estimator framework that smoothly dampens the effect of outliers in the estimation process. This choice was primarily motivated by the simplicity at

which it can be integrated into a nonlinear optimization method. Related to an aforementioned point, whenever the initial guess is not close enough to a local minimum, the M-estimator may incorrectly consider data points as being outliers and thus dramatically bias the optimization. We may address this issue by adapting the M-estimator at each iteration, *i.e.*, being conservative at the beginning and stricter as the algorithm converges to a local minimum. If the sensor data is absolutely not normally distributed, our method may fail and would require an ad-hoc modification, *e.g.*, accounting for multimodal distributions.

5.2.2 Future Work

In addition to the ideas proposed in the above discussion, we can elaborate several lines of research to further develop our calibration algorithm. In the context of autonomous and fully unsupervised systems, it would be particularly relevant to investigate the possibility to automatically guide the platform towards actions that favor observability of the calibration parameters. Indeed, normal operation of the system might not always render the parameters fully observable. An entropy-based strategy may offer the starting point to achieve this goal. In the same direction, as our algorithm requires an expert to a priori design a forward model for each sensor, a fully unsupervised method should be able to bootstrap itself and learn sensor models while acquiring data. To this end, we would delve into the literature of structure learning.

To conclude this part on further work, we shall also briefly mention the future applications of our algorithm. In Chap. 4, we have considered three separate applications that we envision to merge. Our mid-term goal is to perform the full calibration of the sensors mounted on our Toyota Prius, including a Velodyne lidar HDL-64E, 8 MATRIX VISION cameras, an Applanix POS LV 220 GNSS/INS, and the odometry data from the native CAN bus. To achieve this task, we shall further estimate the time delays between the different sensors. Another application currently under investigation concerns the deployment of our algorithm into a pedestrian robot in the context of the European project EUROPA in collaboration with different institutes. The platform is equipped with multiple cameras and laser sensors that require calibration. The foreseen demonstration will involve a reviewer manipulating the sensors and leave the recalibration of the system to our online algorithm. Finally, we are currently deploying our algorithm in a scenario involving the calibration of vehicle dynamics in order to ease the smooth path planning of an autonomous wheel chair.

5.3 Acknowledgements

The final part of this thesis shall be devoted to acknowledging the persons who have encouraged, helped, and supported me along the sinuous way to this doctoral degree.

First and foremost, I would like to express my sincerest gratitude to my advisor and mentor Prof. Roland Siegwart for his continuous support throughout the course of this thesis, for his patience, motivation, kindness, and enthusiasm.

Besides my principal advisor, I would like to thank the rest of my thesis committee: Prof. Jan Peters and Dr. Paul Furgale.

In addition to being in my thesis committee, I owe a great debt to Dr. Paul Furgale for his invaluable supervision. His guidance and immense knowledge helped me in all the time of research and writing of this thesis.

My sincere thanks and gratitude also go to the actual and former senior researchers in our group: Dr. Rudolph Triebel, Dr. Luciano Spinello, Dr. Dizan Vasquez, Dr. Francis Colas, Dr. Stéphane Magnenat, Dr. Alexis Konstantinos, and Dr. Mike Bosse. Their priceless knowledge, experience, and availability were particularly appreciated.

I would like to express my special appreciation and thanks to Stefan Bertschi for his technical support in setting up the Toyota Prius and EUROPA robot.

I am deeply grateful to Dr. Ralf Kaestner, my closest colleague and friend, for his feedback, advice, and constructive comments during the elaboration of this document. Besides having supported me intellectually and technically throughout my doctoral studies, he has always been a great source of inspiration for his meticulous approach to his work and irresistible desire for perfection which I will never catch up.

Special thanks also go to all my former and actual colleagues for the cheerful atmosphere and inestimable discussions, which made my time in the group an unforgettable experience. To name a few: Dr. Martin Rufli, Dr. Laurent Kneip, Dr. Markus Hoepflinger, Dr. Marco Hutter, Dr. Andreas Breitenmoser, Dr. Jiwon Shin, Dr. François Pomerleau, Hannes Sommer, Ulrich Schwesinger, Stefan Leutenegger, Markus Achtelik, Simon Lynen, Javier Alonso Mora, and many others.

Last, but not least, I need to thank my family and friends for everything. I am deeply indebted to my mother and father, for their unlimited encouragements and support towards this doctoral degree. My heartfelt appreciation also goes to my grandparents for having paved the way to what I became and their unquestioning love. I would like to thank my brother for his patience, attention, and kindness to me. Finally, my gratitude goes to all my closest friends, David, Elena, Frédéric, Alain, and Gilbert, who have kindly supported my absence during the past two months and have always been on my side.

Bibliography

[AW80] Murray Aitkin and Granville Tunnicliffe Wilson. Mixture models, outliers, and the EM algorithm. *Technometrics*, 22(3):325–331, 1980.

[BBeM01] Philippe Bonnifait, Pascal Bouron, Paul Crubillé, and Dominique Meizel. Data fusion of four ABS sensors and GPS for an enhanced localization of car-like vehicles. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2001.

[BDW06] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping: Part II. *IEEE Robotics and Automation Magazine (RAM)*, 13(3):108–117, September 2006.

[Ber85] James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 1985.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[BS04] Christopher M. Bishop and Markus Svensen. Robust Bayesian mixture modelling. In *Proc. of the European Symposium on Artificial Neural Networks (ESANN)*, 2004.

[BT11] Jonathan Brookshire and Seth Teller. Automatic calibration of multiple coplanar sensors. In *Proc. of the Robotics: Science and Systems Conference (RSS)*, 2011.

[BT12] Jonathan Brookshire and Seth Teller. Extrinsic calibration from per-sensor egomotion. In *Proc. of the Robotics: Science and Systems Conference (RSS)*, 2012.

[CH92] Tony F. Chan and Per Christian Hansen. Some applications of the rank revealing QR factorization. *SIAM Journal on Scientific and Statistical Computing*, 13(3):727–741, 1992.

[Cha87] Tony F. Chan. Rank revealing QR factorizations. *Linear Algebra and its Applications*, 88-89:67–82, 1987.

[Cha09] Guy Chavent. *Nonlinear Least Squares for Inverse Problems: Theoretical Foundations and Step-by-Step Guide for Applications*. Springer, 2009.

[CM09] Stephen L. Campbell and Carl D. Meyer. *Generalized Inverses of Linear Transformations*. Society for Industrial and Applied Mathematics (SIAM), 2009.

[CMO08] Andrea Censi, Luca Marchionni, and Giuseppe Oriolo. Simultaneous maximum-likelihood calibration of odometry and sensor parameters. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[CS00] Liyu Cao and Howard Schwartz. A directional forgetting algorithm based on the decomposition of the information matrix. *Automatica*, 36(11):1725–1731, 2000.

[Dav05] Andrew J. Davison. Active search for real-time vision. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2005.

[Dav06] Timothy A. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics (SIAM), 2006.

[Dav11] Timothy A. Davis. Algorithm 915: SuiteSparseQR: Multifrontal multithreaded rank-revealing sparse QR factorization. *ACM Transactions on Mathematical Software*, 38(1):1–24, 2011.

[Dev86] Luc Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, 1986.

[DWB06] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics and Automation Magazine (RAM)*, 13(2):99–108, June 2006.

[EM06] Magnus Evestedt and Alexander Medvedev. Stationary behavior of an anti-windup scheme for recursive parameter estimation under lack of excitation. *Automatica*, 42(1):151–157, 2006.

[EM09] Magnus Evestedt and Alexander Medvedev. Elementwise decoupling and convergence of the Ricatti equation in the SG algorithm. *Automatica*, 45(6):1524–1529, 2009.

[EMM04] Yaakov Engel, Shie Mannor, and Ron Meir. The kernel recursive least-squares algorithm. *IEEE Transactions on Signal Processing*, 52(8):2275–2285, 2004.

[FBS12] Paul Furgale, Thimoty D. Barfoot, and Gabe Sibley. Continuous-time batch estimation using temporal basis functions. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[Fio01] Paul D. Fiore. Efficient linear solution of exterior orientation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(2):140–148, 2001.

[Fis22] Ronald Aylmer Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society A*, 222:594–604, 1922.

[Fis25] Ronald Aymler Fisher. *Statistical Methods for Research Workers*. Cosmo Publications, 1925.

[Fis35] Ronald Aymler Fisher. *The Design of Experiments*. Hafner Publishing Company, 1935.

[FKY81] T.R. Fortescue, L.S. Kershenbaum, and B.E. Ydstie. Implementation of self-tuning regulators with variable forgetting factors. *Automatica*, 17(6):831–835, 1981.

[FLM92] Olivier D. Faugeras, Quang-Tuan Luong, and Stephen J. Maybank. Camera self-calibration: theory and experiments. In *Proc. of the European Conference on Computer Vision (ECCV)*, 1992.

[Fur11] Paul Furgale. *Extensions to the Visual Odometry Pipeline for the Exploration of Planetary Surfaces*. PhD thesis, University of Toronto, 2011.

[GCSR03] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2003.

[GKS76] Gene H. Golub, Viriginia Klema, and George W. Stewart. Rank degeneracy and least squares problems. Technical report, Computer Science Department, Stanford University, 1976.

[GL96] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[GM87] Stuart Geman and Donald E. McClure. Statistical methods for tomographic image reconstruction. In *Proc. of the 46th Session of the International Statistical Institute, Bulletin of the ISI*, volume 52, 1987.

[GN88] Alan George and Esmond Ng. On the complexity of sparse QR and LU factorization of finite-element matrices. *SIAM Journal on Scientific and Statistical Computing*, 9(5):849–861, 1988.

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

[GS10] Chao Gao and John R. Spletzer. On-line calibration of multiple LIDARs on a mobile vehicle platform. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2010.

[Han87] Per Christian Hansen. The truncated SVD as a method for regularization. *BIT Numerical Mathematics*, 27(4):534–553, 1987.

[Han88] Per Christian Hansen. The 2-norm of random matrices. *Journal of Computational and Applied Mathematics*, 23(1):117–120, 1988.

[Han98] Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. Society for Industrial and Applied Mathematics (SIAM), 1998.

[Hig96] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics (SIAM), 1996.

[HLP13] Lionel Heng, Bo Li, and Marc Pollefeys. CamOdoCal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[HS88] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proc. of the 4th Alvey Vision Conference*, 1988.

[Hub64] Peter Jost Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[Hub81] Peter Jost Huber. *Robust Statistics*. Jon Wiley & Sons, 1981.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[IKP11] I.C.F. Ipsen, C.T. Kelley, and S.R. Pope. Rank-deficient nonlinear least squares problems and subset selection. *SIAM Journal on Numerical Analysis*, 49(3):1244–1266, 2011.

[Jau07] Claude Jauffret. Observability and Fisher information matrix in nonlinear regression. *IEEE Transactions on Aerospace and Electronic Systems*, 43(2):756–759, 2007.

[Jay57] Edwin Thompson Jaynes. Information theory and statistical mechanics. *The Physical Review*, 106(4):620–630, 1957.

[Jaz70] Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[Jef46] Harold Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 186(1007):453–461, 1946.

[Kal60] Rudolf Kalman. On the general theory of control systems. In *Proc. of the First IFAC Congress Automatic Control*, 1960.

[KC06] Jae-Hean Kim and Myung Jin Chung. Absolute motion and structure from stereo image sequences without stereo correspondence and analysis of degenerate cases. *Pattern Recognition*, 39:1649–1661, 2006.

[Kel99] C.T. Kelley. *Iterative Methods for Optimization*. Society for Industrial and Applied Mathematics (SIAM), 1999.

[KGB11] Rainer Kuemmerle, Giorgio Grisetti, and Wolfram Burgard. Simultaneous calibration, localization, and mapping. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011.

[KL51] Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[Kol50] Andrei Nikolajevich Kolmogorov. *Foundations of the Theory of Probability*. Chelsea Publishing Company, 1950.

[KS11] Jonathan Kelly and Gaurav S. Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *The International Journal of Robotics Research (IJRR)*, 30(1):56–79, 2011.

[KS12a] Alonzo Kelly and Neal Seegmiller. A vector algebra formulation of mobile robot velocity kinematics. In *Proc. of the International Conference on Field and Service Robotics (FSR)*, 2012.

[KS12b] Henrik Kretzschmar and Cyrill Stachniss. Information-theoretic pose graph compression for laser-based SLAM. *International Journal of Robotics Research (IJRR)*, 31(11):1219–1230, 2012.

[Lin56] Dennis Victor Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):879–1212, 1956.

[LRAAD10] Pierre Lébraly, Eric Royer, Omar Ait-Aider, and Michel Dhome. Calibration of non-overlapping cameras - application to vision-based robotics. In *Proc. of the British Machine Vision Conference (BMVC)*, 2010.

[LT10] Jesse Levinson and Sebastian Thrun. Unsupervised calibration for multi-beam lasers. In *Proc. of the International Symposium on Experimental Robotics (ISER)*, 2010.

[Mar63] Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, 11(2):431–441, 1963.

[MBL89] Chia-Hsiang Menq, Jin-Hwan Born, and Jim Z. Lai. Identification and observability measure of a basis set of error parameters in robot calibration. *Journal of Mechanisms, Transmissions, and Automation in Design*, 111:513–518, 1989.

[MF92] Stephen J. Maybank and Olivier D. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision (IJCV)*, 8(2):123–151, 1992.

[MFS13] Jerome Maye, Paul Furgale, and Roland Siegwart. Self-supervised calibration for robotic systems. In *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*, 2013.

[MHN12] Will Maddern, Alastair Harrison, and Paul Newman. Lost in translation (and rotation): Rapid extrinsic calibration for 2D and 3D LIDARs. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[MR07] Christopher Mei and Patrick Rives. Single view point omnidirectional camera calibration from planar grids. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[MR08] Faraz M. Mirzaei and Stergios I. Roumeliotis. A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation. *IEEE Transactions on Robotics*, 24(5):1143 – 1156, 2008.

[MSS06] Agostino Martinelli, Davide Scaramuzza, and Roland Siegwart. Automatic self-calibration of a vision system during robot motion. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[Mur12] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[Ney37] Jerzy Neyman. Outline of a theory of statistical estimation based on the classical theory of probability. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 236(767):333–380, 1937.

[NTP11] Duy Nguyen-Tuong and Jan Peters. Incremental online sparsification for model learning in real-time robot control. *Neurocomputing*, 74(11):1859–1867, 2011.

[Pea00] Karl Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5*, 50(302):157–175, 1900.

[Pow70] Michael J.D. Powell. A new algorithm for unconstrained optimization. In J.B. Rosen, O.L. Mangassarian, and K. Ritter, editors, *Nonlinear Programming*, pages 31–65. Academic Press, 1970.

[QSS07] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical Mathematics*. Springer, 2007.

[Rao45] Calyampudi Radakrishna Rao. Information and the accuracy attainable in the estimation of statistical parameters. *Bulletin of the Calcutta Mathematical Society*, 37:81–89, 1945.

[Rot71] Thomas J. Rothenberg. Identification in parametric models. *Econometrica*, 39(3):577–591, 1971.

[RSS08] Martin Rufli, Davide Scaramuzza, and Roland Siegwart. Automatic detection of checkerboards on blurred and distorted images. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

[RW06] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[SEF85] Steinar Saelid, Olav Egeland, and Bjarne Foss. A solution to the blow-up problem in adaptive controllers. *Modeling, Identification and Control*, 6(1):39–56, 1985.

[SG00] Chris Stauffer and W. Eric L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(8):747–757, August 2000.

[SGB05] Cyrill Stachniss, Giorgio Grisetti, and Wolfram Burgard. Information gain-based exploration using Rao-Blackwellized particle filters. In *Proc. of the Robotics: Science and Systems Conference (RSS)*, 2005.

[Sha48] Claude Elwood Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[SHN10] Mark Sheehan, Alastair Harrison, and Paul Newman. Automatic self-calibration of a full field-of-view 3D n-laser scanner. In *Proc. of the International Symposium on Experimental Robotics (ISER)*, 2010.

[SHN12] Mark Sheehan, Alastair Harrison, and Paul Newman. Self-calibration for a 3D laser. *The International Journal of Robotics Research (IJRR)*, 31(5):675–687, 2012.

[SM99] Peter F. Sturm and Stephen J. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.

[SS11] Matthew S. Shotwell and Elizabeth H. Slate. Bayesian outlier detection with Dirichlet process mixtures. *Bayesian Analysis*, 6(4):1–22, 2011.

[SZS+08] R.S. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(6):1068–1080, 2008.

[Tar05] Albert Tarantola. *Inverse Problem Theory and Methods for Model Parameter Estimation*. Society for Industrial and Applied Mathematics (SIAM), 2005.

[TDS07] Jo-Anne Ting, Aaron DSouza, and Stefan Schaal. Automatic outlier detection: A Bayesian approach. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[Tew76] Reginald P. Tewarson. *Sparse Matrices*. Academic Press, 1976.

[TFB12] Chi Hay Tong, Paul Furgale, and Timothy D. Barfoot. Gaussian process Gauss-Newton: Non-parametric state estimation. In *Proc. of the Conference on Computer and Robot Vision (CRV)*, 2012.

[UHS07] James Underwood, Andrew Hill, and Steve Scheding. Calibration of range sensor pose on mobile platforms. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[vdS69] A. van der Sluis. Condition numbers and equilibration of matrices. *Numerische Mathematik*, 14(1):14–23, 1969.

[Wu81] Chien-Fu Wu. Asymptotic theory of nonlinear least squares estimation. *The Annals of Mathematical Statistics*, 9(9):501–513, 1981.

[XC06] Yang Xie and Bradley P. Carlin. Measures of Bayesian learning and identifiability in hierarchical models. *Journal of Statistical Planning and Inference*, 136(10):3458–3477, 2006.

[Yan81] Mihalis Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 2(1):77–79, 1981.

[Zha99] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 1999.

[ZP04] Qilong Zhang and Robert Pless. Extrinsic calibration of a camera and laser range finder (improves camera calibration). In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004.

Index

Fuchsiastrasse 15
8048 Zürich, Switzerland
+41-(78)-789-4284
+41-(44)-632-8987
jerome.maye@mavt.ethz.ch

Jérôme Maye

Curriculum Vitae

Personal Information

Date of birth	11/13/1981
Marital status	Unmarried

Education

Since 05/2009 **Studies towards a Doctor of Sciences in Robotics, Systems, and Control**, *ETH Zurich*, Switzerland, Graduation expected in 04/2014.
Thesis: "Online Self-Calibration for Robotic Systems"

03/2007 **Msc Computer Science**, *Ecole Polytechnique Fédérale de Lausanne*, Switzerland.
Thesis: "Control of Locomotion in Modular Robotics", Specialization: Computer Engineering

06/2001 **Scientific Maturity**, *Lycée-Collège des Creusets, Sion*, Switzerland.

Working Experience

07/2008 – **Research assistant**, *ETH Zurich*, Switzerland.
Student projects supervision, European and ESA projects, software and hardware engineering

11/2007 – 06/2008 **Embedded software developer**, *Atracsys LLC*, Lausanne, Switzerland.
Development of 3D tracking systems

05/2007 – 10/2007 **Research assistant**, *Ecole Polytechnique Fédérale de Lausanne*, Switzerland.
Development of 3D tracking systems

07/2006 – 10/2006 **Internship**, *Ecole Polytechnique Fédérale de Lausanne*, Switzerland.
Development of an FPGA-based development board for students (Fpga4u)

03/2006 – 06/2006 **Internship**, *Ecole Polytechnique Fédérale de Lausanne*, Switzerland.
Construction and testing of modular robots (YaMoR II)

Extracurricular Activities

Since 07/2008 **Web design and back-end development, server administration**, *Autonomous Systems Lab, ETH Zurich*, Switzerland.
Complete list of projects: https://github.com/jmaye

09/2010 **Smart Urban Stage**, Zurich, Switzerland, Winner of the "Smart Future Minds" award (CHF 15'000).
Project title: Future transportation system

01/1992 – **Musician**, Switzerland.
Singer and keyboard player in a band, various operas and musical events

Refereed Publications

J. Maye, P. Furgale, and R. Siegwart, "Self-supervised calibration for robotic systems," in *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*, 2013.

J. Maye, R. Kaestner, and R. Siegwart, "Curb detection for a pedestrian robot in urban environments," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

R. Kaestner, J. Maye, Y. Pilat, and R. Siegwart, "Generative object detection and tracking in 3D range data," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

J. Maye, R. Triebel, L. Spinello, and R. Siegwart, "Bayesian on-line learning of driving behaviors," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

J. Maye, L. Spinello, R. Triebel, and R. Siegwart, "Inferring the semantics of direction signs in public places," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

A. Sproewitz, R. Moeckel, J. Maye, and A. J. Ijspeert, "Learning to move in modular robots using central pattern generators and online optimization," *The International Journal of Robotics Research (IJRR)*, vol. 27, no. 3–4, pp. 423–443, 2008.

R. Moeckel, A. Sproewitz, J. Maye, and A. J. Ijspeert, "An easy to use Bluetooth scatternet protocol for fast data exchange in wireless sensor networks and autonomous robots," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007.