BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF MECHANICAL ENGINEERING
FAKULTA STROJNÍHO INŽENÝRSTVÍ

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS
ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

HYBRID METHOD FOR MODELLING AND STATE ESTIMATION OF DYNAMIC SYSTEMS
HYBRIDNÍ METODA PRO MODELOVÁNÍ A POZOROVÁNÍ STAVŮ DYNAMICKÝCH SYSTÉMŮ

SHORT VERSION OF DOCTORAL TESIS
ZKRÁCENÁ VERZE DOKTORSKÉ PRÁCE

AUTHOR — Ing. Martin Brablc
AUTOR PRÁCE

SUPERVISOR — doc. Ing. Robert Grepl, Ph.D.
ŠKOLITEL

BRNO 2023

Keywords
Dynamic systems; Modelling and simulation; Control theory; Continuous time systems; Discrete time systems; Approximation methods; Local linear models; Local linear regression; Receptive fields; Weighted regression; MATLAB; Parameter estimation; State estimation; Kalman filter; Process noise; Gaussian Noise Covariance; Simultaneous estimation; Joint estimation; Dual estimation; Recursive least squares;

Klíčová slova
Dynamické systémy; Modelování a simulace; Teorie řízení; Systémy se spojitým časem; Systémy s diskrétním časem; Aproximační metody; Lokální lineární modely; Lokální lineární regrese; Receptivní pole; Vážená regrese; MATLAB; Odhadování patametrů; Odhadování stavů; Kalmanův filtr; Procesní šum; Kovariance Gaussovs- kého šumu; Silmultánní estimace, Sloučená estimace; Duální estimace; Rekursivní metoda nejmenších čtverců;

Místo uložení rukopisu:
Knihovna VUT-FSI v Brně, Technická2896/2, 61669 Brno.
Martin Brablc, 2023
ISBN 80-214-
ISSN 1213-4198

Contents

1 Introduction

In this thesis, we deal primarily with systems that are typical for the field of mechatronics. These are systems with concentrated parameters of low order, often nonlinear, which typically originated in several other domains, e.g., electronics, electromagnetics, solid body mechanics, thermo- or hydromechanics, etc. A very common property is the existence of an algebraic or differential relationship between the system states, meaning that we can often find pairs or larger groups of states being in the time domain derivative/integral relation, e.g., position - velocity, charge current, etc., which gives the system special properties that we further investigate. A typical example of such a mechatronic system is an electric drive.

When implementing a control algorithm, especially when dealing with nonlinear systems, there are three distinct tasks which need to be solved:

- creating a model of the system,
- choosing and tuning a signal processing filter or state observer,
- designing n and tuning the control algorithm itself.

Usually, the model of the system we want to control, often called the plant, is required to be able to simulate the system response in various situations and in combination with various filters and controllers without the risk of destroying the real plant. There are many different approaches to modelling dynamic systems, and the choice is often very application-specific. For the purpose of this thesis, we assume that we can deduce a set of ordinary differential equations (ODEs) based on the first-principles approach, which describes or approximates the system at hand with reasonable precision. By reasonable precision, we mean that we can use it to identify the system states and significant nonlinearities, at least on what states the nonlinearities depend on. This set of ODEs can be directly used as a model of the system (assuming that we are able to estimate or measure the unknown parameters) or it can be further approximated by one of many various methods which loosely fall into categories called local and global methods.

The second base task that needs to be dealt with when developing a control algorithm can be summarised as estimating the states of the system to be controlled. This task is important as it provides the necessary feedback information for any control algorithm, and the signal quality (accuracy, delay, signal-to-noise ratio, etc.) greatly influences the quality and stability of the control process. Solutions to this task generally fall into two categories: filters and state estimators. Putting aside obvious low-pass and anti-aliasing filters, derivative filters are often used in situations where one of the states is not measured directly, such as central difference or Savitzky-Golay type filters. For example, when we only measure the position of a mechanism and its speed, which is also one of the system states that needs to be determined using a derivative filter.

The third task is the design of the control algorithm itself. Aside from the common linear control design methods (PID, LQR , etc.), there are many methods and approaches

for the design of a control law for nonlinear dynamic systems. At random, we can mention feedback linearisation, reinforcement learning, model predictive control (MPC), State-Dependent Riccati Equation (SDRE), gain scheduling, and many others. Independent of the method we choose to implement, the control design methods mostly have to assume the knowledge of all the controlled system states. This requirement is often not mentioned when describing the design of complex nonlinear control algorithms and is left for the reader to deal with on their own, based on the specific application. In practice, this presents a problem which is neither trivial nor exact and usually requires the use of one of the above-mentioned state estimation techniques.

To simplify the design process, the three tasks are, in most practical applications, being solved independently. First, the models of the system are created, then filters or state estimators are implemented and tuned in simulation or with the real plant. Lastly, both are used to implement a suitable control algorithm. The design methods for all three tasks have many parameters which need to be set by the developer, while some are not exact, lacking a guide or criterion function to be correctly set. This is the most common, independent design process, even though the results of all the tasks are highly intertwined and, in the end, serve to solve a single criterion, the control process quality.

The problems described above often result in a suboptimal control design and push developers to choose the design method out of the great variety of available algorithms they know and have experience tuning, as implementing a new, previously unknown algorithm requires acquiring enough experience to tune it properly. With that in mind, the main goal of this dissertation is to simplify the control design process by removing, or at least minimising, the need to tune some of the most critical design parameters. Based on previous research and experience, this goal can be achieved by combining independent tasks into a single adaptive method or algorithm, which would only be subject to the overall control quality. We will mainly focus on combining the first two tasks, modelling and state estimation, as these have practical applications on their own and may serve as a basis for future research in combining all three tasks.

Our goal is to develop a hybrid method for the simultaneous estimation of states and parameters of nonlinear dynamic systems, which would adapt, online, the model of the system and tune a state estimator as new data become available. It is an approach similar to dual estimation, where one of the filters is replaced by a more complex approximation method, preferably based on local linear models. Both parallel algorithms share not only their best estimates, but also confidence in that estimate, which allows the automatic parameter tuning.

2 Theoretical Survey

2.1 Dynamic systems

Under the term *dynamic system*, we can imagine a somewhat simplified representation (model) of a real system that we have in mind, specifically a system that evolves over time. By "simplified", we mean that we only observe the outward behaviour of the system and the specific physical quantities we want to observe or predict, whose model is usually in the form of a set of differential equations. The level of detail is highly dependent on the actual application. In the literature, for example, in [1, 2], we can find various mathematical definitions of a dynamic system, however, all of them define it using similar terms: a *state*, a *phase/state space* and an *evolution function*.

Depending on the type of the set of all possible time instants T, a system can be in, we can differentiate dynamic systems into *continuous* in time ($\mathrm{T} \subseteq \mathbb{R}$) and *discrete* in time ($\mathrm{T} = \{kT_s : k \subseteq \mathbb{Z}\}$, where T_s is a *sampling period*).

Then, for both discrete and continuous systems, the *evolution function* (2.1)

$$\phi\left(t\,, t_0\,, \boldsymbol{x}_0\right) : \mathrm{T} \times \mathrm{T} \times \Omega \to \Omega \tag{2.1}$$

unequivocally describes the state $\boldsymbol{x}_t \in \Omega$ of the system at time instant $t \in \mathrm{T}$ defined by the initial state $\boldsymbol{x}_0 \in \Omega$ at time instant $t_0 \in \mathrm{T}$.

Usually, when describing a dynamic system, we start from a set of first-order differential equations (2.2),

$$\dot{\boldsymbol{x}}(t) = f\left(\boldsymbol{x}(t)\,, t\right) \tag{2.2}$$

where $\dot{\boldsymbol{x}}(t) \in \mathbb{R}^n$ is a vector of the time domain derivatives of each of the state quantities contained in $\boldsymbol{x}(t)$ and f in (2.3),

$$f\left(\boldsymbol{x}(t), t\right) : \Omega \times \mathrm{T} \to \mathbb{R}^n \tag{2.3}$$

where f is a vector-valued function that describes the dynamics of the system. Then, the evolution function (2.1) is the solution to the *initial value problem* (IVP) of the set of equations (2.3) [1].

In this thesis with systems that use $\mathrm{T} = \{kT_s : k \in \mathbb{Z}^+\}$, we will denote a state $\boldsymbol{x}$ at a time instant $t_k \in T$ by a subscript index k, i.e., $\boldsymbol{x}_{t_k} = \boldsymbol{x}_k$.

It is useful to develop the evolution function for this special case of a system with a discrete set T to be able to make a transformation between two consecutive time instants as in (2.4).

$$\boldsymbol{x}_k = \phi\left(t_k\,, t_{k-1}\,, \boldsymbol{x}_{k-1}\right) \tag{2.4}$$

We will refer to this special case of the evolution function as *evolution operator* $\Phi : \Omega \times \mathrm{T} \to \Omega$. Then, we can write a discrete model of an autonomous dynamic system as (2.5).

$$\boldsymbol{x}_k = \Phi\left(\boldsymbol{x}_{k-1}, t_{k-1}\right) \tag{2.5}$$

In practice, we often deal with systems that are not autonomous and have an external input. For this reason, we must extend our description of dynamic systems by a set of all possible inputs $\Psi \subseteq \mathbb{R}^m$, where m is the number of external inputs and $\boldsymbol{u}(t) \in \Psi$ is a specific set of inputs applied to the system at any given time instant $t \in T$. Then, we can expand the models (2.2) and (2.5) by an additional argument, acquiring a model of the dynamics of a *controlled* system (2.6). Also, to further simplify the notation, we will be marking arguments which are functions of time, e.g. $\boldsymbol{x}(t)$, simply as $\boldsymbol{x}$ as the time dependence is obvious in most cases.

$$\dot{\boldsymbol{x}} = f\left(\boldsymbol{x}, \boldsymbol{u}, t\right) \tag{2.6}$$

We can also define the evolution operator for a discrete system with an input as (2.7).

$$\boldsymbol{x}_k = \Phi\left(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}, t_{k-1}\right) \tag{2.7}$$

Together with the basic system models and dynamic systems themselves, there are a number of additional models being used for convenience in practice. Often, it is useful to separate the output of the system from its state. Generally, it can be described by Equation (2.8).

$$\boldsymbol{y} = g\left(\boldsymbol{x}, \boldsymbol{u}, t\right) \tag{2.8}$$

where $\boldsymbol{y}$ is the output of the system corresponding to state $\boldsymbol{x}$ and input $\boldsymbol{u}$. Often, the output is simply a subset of state $\boldsymbol{y} \subseteq \boldsymbol{x}$. The output often represents one or more quantities that we are interested in tracking or controlling.

Furthermore, we can also use a measurement model to link the quantities that we actually measure with the state, using Equation (2.9).

$$\boldsymbol{z} = h\left(\boldsymbol{x}, t\right) \tag{2.9}$$

where $\boldsymbol{z}$ is the measurement vector corresponding to state $\boldsymbol{x}$. The measurement can also be a subset of the state vector $\boldsymbol{z} \subseteq \boldsymbol{x}$.

A very important special case of all the above mentioned models is a linear system. Assuming that the system at hand is linear, time-invariant, and continuous, the general model of the system dynamics (2.6), the model of the output (2.8), and the model of the measurement (2.9), can be transformed into linear case equations (2.10), (2.11), and (2.12), respectively, forming the so-called state space model of a dynamic system.

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{2.10}$$

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u} \tag{2.11}$$

$$\boldsymbol{z} = \boldsymbol{H}\boldsymbol{x} \tag{2.12}$$

Similarly, for linear time-invariant discrete systems, the discrete dynamic model (2.7) with input and measurement models can be transformed into (2.13), (2.14), and (2.15), respectively, forming the so-called discrete state space model of a dynamic system.

$$\boldsymbol{x}_k = \boldsymbol{F}\boldsymbol{x}_{k-1} + \boldsymbol{G}\boldsymbol{u}_{k-1} \tag{2.13}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}\boldsymbol{u}_k \tag{2.14}$$

$$\boldsymbol{z}_k = \boldsymbol{H}\boldsymbol{x}_k \tag{2.15}$$

Lastly, by adding the assumption that the model of a continuous system (2.6) and a discrete system (2.7) is not dependent on time, but $\boldsymbol{F}$ and Φ depend on a set of parameters $\boldsymbol{b} \in \mathbb{R}^p$ instead, where p is the number of parameters, both can be rewritten in the form of (2.16) and (2.17), respectively. Both will be useful in the later chapters.

$$\dot{\boldsymbol{x}} = f\left(\boldsymbol{x}\,, \boldsymbol{u}\,, \boldsymbol{b}\right) \tag{2.16}$$

$$\boldsymbol{x}_k = \Phi\left(\boldsymbol{x}_{k-1}\,, \boldsymbol{u}_{k-1}\,, \boldsymbol{b}_{k-1}\right) \tag{2.17}$$

Also, when talking about parameters, it is useful to note a special case of models *linear in parameters*. This is a weaker condition than a linear system, it allows for nonlinearities, but the parameter $\boldsymbol{b}$ must only occur as an argument of the linear functions. [3]

This property can be defined by fulfilling the condition (2.18). A model of a dynamic system $\boldsymbol{F}\left(\boldsymbol{x}\,, \boldsymbol{u}\,, \boldsymbol{b}\right)$ is linear in parameters exactly when

$$\frac{\partial f\left(\boldsymbol{x}\,, \boldsymbol{u}\,, \boldsymbol{b}\right)}{\partial b_i} \neq \boldsymbol{0} \wedge \frac{\partial^2 f\left(\boldsymbol{x}\,, \boldsymbol{u}\,, \boldsymbol{b}\right)}{\partial b_i^2} = \boldsymbol{0}\,, \forall b_i \in \boldsymbol{b} \tag{2.18}$$

2.2 Kalman filter

The *Kalman filter* was developed by several mathematicians and physicists in the late 1950s and is named after the Hungarian mathematician and engineer Rudolf E. Kálmán, who published his now famous paper [4] in 1960. In the common implementation, the Kalman filter (KF) is based on a model of a linear discrete dynamic system with continuous state space, which can be described by (2.13) and a model of the measurement process as described by (2.15). Both of these models are assumed to be imprecise, and the imprecision is modelled by Gaussian zero-mean noise. Specifically, a noise term is added to both models as in (2.19) and (2.20),

$$\boldsymbol{x}_k = \boldsymbol{F}_k \boldsymbol{x}_{k-1} + \boldsymbol{G}_k \boldsymbol{u}_{k-1} + \boldsymbol{w}_k \tag{2.19}$$

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k \tag{2.20}$$

where $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{Q}_k, 0)$ represents the *process noise* term and $\boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{R}_k, 0)$ represents the *measurement noise* term, where $\boldsymbol{Q}_k = cov(\boldsymbol{w}_k)$ is the process noise covariance and $\boldsymbol{R}_k = cov(\boldsymbol{v}_k)$ is the measurement noise covariance.

Various notations are used to describe KF. In this thesis, we will use the hat symbol over a variable symbol (e.g. $\hat{\boldsymbol{x}}$) to mark that it is an estimate of the true value and expand the lower indexing notation by the number of the iteration the estimate is based on. For example, $\hat{\boldsymbol{x}}_{i|j}$ represents an estimate of the value of $\boldsymbol{x}$ in iteration i based on information from iteration j. Typically, we use either information from the previous iteration ($j = i - 1$), which is called the *a priori* or the *prior* estimate, or information from the actual iteration ($i = j$), which is called the *a posteriori* or the *posterior* estimate.

Then, every state estimate that we work with is a tuple (2.21),

$$\left(\hat{\boldsymbol{x}}_{i|j},\, \hat{\boldsymbol{P}}_{i|j}\right) \tag{2.21}$$

where $\hat{\boldsymbol{x}}_{i|j}$ is the estimate of the state vector $\boldsymbol{x}_i$ and $\hat{\boldsymbol{P}}_{i|j}$ is the estimate of the covariance matrix $\boldsymbol{P}_i$ that describes the confidence of the state estimate in the form of a Gaussian *probability density function* (PDF). We can also say that the true state position in the state space is described by $\boldsymbol{x_i} \sim \mathcal{N}(\hat{\boldsymbol{P}}_{i|j},\, \hat{\boldsymbol{x}}_{i|j})$ with $\mathcal{N}$ as the PDF. To simplify the notation, we will drop the hat symbol on the covariance matrix estimate and simply refer to it as $\boldsymbol{P}_{i|j}$, as is common in most texts, as we never deal with true covariance, only with its estimate, making the estimate symbol obsolete.

Specifically, the Kalman filter algorithm is as follows. We start with an initial guess of the state estimate and the covariance matrix ($\hat{\boldsymbol{x}}_{0|0},\, \boldsymbol{P}_{0|0}$). Then, in every iteration k of the tracking process (usually every time we get a new measurement), we perform the *prediction step* to acquire the *a priori* estimate of the state vector and its covariance according to (2.22) and (2.23).

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{P}_k \hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{G}_k \boldsymbol{u}_{k-1} \tag{2.22}$$

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^T + \boldsymbol{H}_k \tag{2.23}$$

After that, we use the measurement z_k to perform *the update step* to acquire the *a posteriori* estimates. First, we calculate the Kalman gain $\boldsymbol{K}$ according to (2.24) and then perform the updates (2.25) and (2.26), where I is the identity matrix of the appropriate order.

$$\boldsymbol{H}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T \left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T + \boldsymbol{R}_k\right)^{-1} \tag{2.24}$$

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{H}_k \left(\boldsymbol{z}_k - \boldsymbol{H}_k \hat{\boldsymbol{x}}_{k|k-1}\right) \tag{2.25}$$

$$\boldsymbol{P}_{k|k} = \left(I - \boldsymbol{H}_k \boldsymbol{H}_k\right) \boldsymbol{P}_{k|k-1} \tag{2.26}$$

After the update, we acquire the new estimates ($\hat{\boldsymbol{x}}_{k|k}$, $\boldsymbol{P}_{k|k}$), which can be used again in the next iteration $k+1$.

Generally, correctly setting both the process and the measurement noise covariances is the main problem when implementing KF in a practical implementation. The measurement noise covariance $\boldsymbol{R}$ can usually be measured, estimated, guessed, or read from a sensor datasheet. Correctly setting the process noise covariance $\boldsymbol{Q}$ is another issue, and the question *How to set the process noise covariance for a Kalman filter?* is probably one of the most googled questions by students, engineers, and researchers implementing their first filters [5].

We can find several *ad hoc* methods to set $\boldsymbol{Q}$ in some special cases where the perfect Kalman conditions are met, for example in [5, 4], also several adaptive autotuning algorithms were proposed, for example in [6, 7, 8, 9, 10]. However, in most practical applications, the $\boldsymbol{Q}$ matrix is set experimentally by intuition.

2.3 Simultaneous Estimation

The purpose of *simultaneous estimation* is, in the course of the measurement process, striving to estimate both the states and the parameters of a model of a dynamic system at the same time. Assuming a model in the form of (2.16) or (2.17), we target the state vector $\boldsymbol{x}_k$ and the parameters $\boldsymbol{b}_k$, iteratively estimating them as best as possible based on the known input u_k and the measurement vector z_k. We also consider the structure of the model, the function f for continuous or the Φ for discrete systems, to be completely known.

Typically, a suitable variant of the Kalman filter is used for the estimation of the state vector, outputting the estimate of the state vector $\hat{x}_{k|k}$ together with its covariance matrix $\boldsymbol{P}_{k|k}$, as described in Section 2.2 or, in more detail, regarding the choice of the most suitable method for a specific task, in [11, 12].

There are many various methods for *parameter estimation* (PE), which would iteratively adjust the estimate based on new data points (online). In case of a system which is linear or linear in parameters, for example, the recursive least squares method (RLS) can be used (see [3] for further details on RLS and other PE methods).

In the case where we need to estimate both the states and the parameters, there are two distinct approaches that can be used. These are *joint estimation* and *dual estimation*, which we outline in further detail in the subsequent sections. Regardless of the chosen approach, all the applications utilise some of the Kalman filter variants in different configurations. Common characteristics, typical for the KF, also follow from this. It is always necessary to correctly set the critical tuning parameters, especially the process noise covariance.

2.3.1 Joint estimation

The simplest way to implement *simultaneous estimation* is *joint estimation*. As mentioned in the Introduction, while using this approach, we consider the parameters of the dynamic system to be estimated as constant states. This means that we can create an extended state vector $\overline{\boldsymbol{x}}_k = \left[\boldsymbol{x}_k^T, \boldsymbol{b}_k^T\right]^T$ and reformulate the initial models (2.16) and (2.17) into (2.27) and (2.28), respectively.

$$\dot{\overline{\boldsymbol{x}}}_k = f\left(\overline{\boldsymbol{x}}_k\,, \boldsymbol{u}_k\right) \tag{2.27}$$

where $\dot{\overline{\boldsymbol{x}}}_k = \left[\dot{\boldsymbol{x}}_k^T\,, \mathbf{0}\right]^T$.

$$\overline{\boldsymbol{x}}_k = \Phi\left(\overline{\boldsymbol{x}}_{k-1}\,, \boldsymbol{u}_{k-1}\right) \tag{2.28}$$

Similarly to any other system, we can use the KF to estimate the state vector of the system (2.27) or (2.28). The state extension, in most cases, causes the system to be highly nonlinear, which forces us to use one of the nonlinear Kalman filter versions. The nonlinearity occurs even if the initial system was linear, as the parameters now considered states usually are in product with the original states, causing a significant nonlinearity.

2.3.2 Dual estimation

The second method, which is used more frequently, is *dual estimation*. In this variant, we do not alter the system model or state vector in any way, but implement two parallel Kalman filters. The first filter estimates the system states as usual. The second filter is based on the same system model as the first, but we swap what we consider states and parameters, i.e. the parameters are considered constant states, and the states are considered parameters that vary in time. In every iteration, both filters exchange information about their best estimates of the current states and parameters. Typically, two identical filters (EKF or UKF) are used.

A significant advantage of the n is the relatively lower computational complexity, thanks to working with systems of lower order and better numerical stability, avoiding the issues of joint estimation. However, there may be a disadvantage in not considering correlations between the estimates of the states and the parameters, which may cause slower or biased convergence or even instability, as was suggested in [13].

2.4 Model approximation methods

When we work with a dynamic system in the form of (2.6), there is a question of finding the structure of the function f based on the measured data. Generally, this task falls into the field of *dynamic system identification* [3, 14].

In a case where acquiring the function based on the knowledge of the system using one of the general analytical modelling methods is not possible, we may choose to use one of many general approximation methods to construct a fitting model of the function based on measured data, usually the inputs and outputs of the system. As we mentioned earlier, these methods can be divided into *local* and *global* approximation methods. In the context of machine learning, these groups are called *lazy learning* and *eager learning* [3], where the difference is described based on the willingness of the method to generalise new information. Generally, we can say that local (lazy) methods are better suited for applications where we require adaptiveness and constantly acquire new data across the entire reachable state space of the system, which is exactly the situation that we are dealing with. The other advantages of local approximation methods may be the higher

stability and the sometimes easier interpretation for humans.

These are the reasons that we decided to mainly use local approximation methods, specifically local linear models as these bring other benefits which will become clearer in later chapters.

Figure 2.1 demonstrates the approximation of a one-dimensional nonlinear function using local linear modes with Gaussian weight functions.

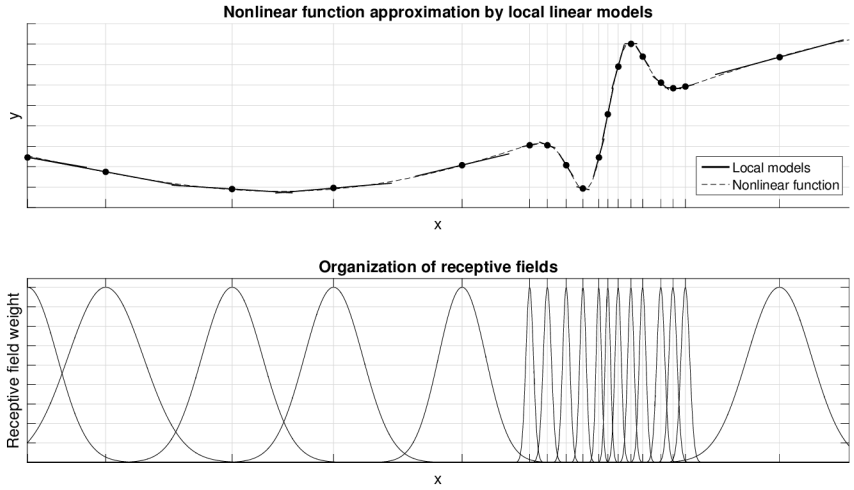

Figure 2.1: An example of approximation of a one-dimensional nonlinear function using local linear models with Gaussian weight functions using the RFWR algorithm. Taken from [15].

We will focus on the RFWR method with the further experiments. The original algorithm was first presented in [16] and then expanded for control applications and higher-dimensional problems in [17].

A significant advantage of the RFWR method is that, thanks to the gradient optimisation of the receptive fields, it has the ability to approximate complex shapes using a relatively low number of local models. However, it has several tuning parameters which need to be set correctly, otherwise the approximation process may be unstable, imprecise, or on the contrary lead to overfitting.

A detailed description of the algorithm can be found in [16, 18, 19].

3 Formulation of the thesis goals

The significance and need for the simultaneous estimation are outlined in Chapter 1 in relation to the typical control design process and related tasks. In Chapter 2, we described several existing variants of methods for the individual as well as the simultaneous estimation of states and parameters of dynamic systems, and local linear models of dynamic systems.

These methods are functional, however, none of them are universally applicable and there is no clear metric or methodology for the correct method choice. In practice, developers and scientists often choose the approach with which they have the most experience, regardless of the other methods. The existing methods also have significant disadvantages as described in Chapter 2, especially requiring intuition-based tuning of many application specific parameters, the implementation is often very layman unfriendly and deals with the estimation of the states and parameters (in terms of parameter tuning criteria) separately, even though they are closely intertwined.

Based on previous experiments and research, the goal of this thesis is to contribute to the solution of the dual estimation problem by developing a hybrid dual estimation method based on local approximation (RFWR for example) for the parameter estimation and the Kalman filter for the state estimation, instead of using the same algorithm for both estimation tasks. One of the main issues to deal with is the correct (preferably automated or adaptive) tuning of the Kalman filter's process noise covariance matrix so that the dual estimation is stable and performs well, at least for a specific, limited case of a typical mechatronic system, defined in Chapter 1.

It is important to note that we are not primarily aimed at improving the overall estimation quality, but to simplify the implementation process while keeping the quality good enough. That being said, however, improving the quality of the estimation will be a secondary target.

Specifically, we defined four goals for this research, which correlate with the goals set and approved at the State doctoral exam:

Survey and research of the mutual relationship of typical state and parameter estimation methods for dynamic systems

Chapter 2 describes the relationship of two common tasks that often occur while working with nonlinear dynamic systems - state estimation and parameter estimation. The first part of the thesis is to research, in detail, the most common methods and algorithms used for both these tasks with the goal of finding and exploring the possibilities of their mutual interconnection. We will mainly deal with methods from the Bayesian recursive filter category (variants of the Kalman filter) for the state estimation task and with local linear approximation methods (LWL, RFWR, LOLIMOT, LWPR, etc.) for the modelling and parameter estimation task.

The interconnection of both tasks can be found, for example, in the imprecision or uncertainty of the estimation of parameters, which also translates into the states and

output of the system. This influences the process noise of the system model, which is one of the most important effects that affects the implementation of the Kalman filter or variants. Some parameter estimation methods (such as Recursive least squares (RLS)) directly deal with the parameter estimation imperfection model in the form of Gaussian noise uncertainty. Specifically, questions arise concerning what is the effect of the known parameter uncertainty covariance on the process noise covariance in case of a linear system or what other effects influence it.

The output of this step will be an analysis of the signal and parameters occurring in both tasks and determining which of them may be used to better tie the said algorithms together.

Modification of the RFWR method for its use on typical mechatronic systems

The original algorithm described in [16] was successfully used in several practical applications, however, it almost exclusively involves offline data processing. As Chapter 2 describes its potential in local linear approximation, the original algorithm has several disadvantages that limit its capabilities when applied in the context of the three tasks leading to the control design, especially on typical mechatronic systems.

Namely, these are:

- problematic stability while incrementally adapting to new data, e.g. when the data is asymmetrically localised with respect to the local model centre,
- vulnerability to overly local data,
- the validity function dimension is determined by the order of the system,
- possible inconsistencies between neighbouring local models.

The goal of this thesis is to modify the original RFWR algorithm in a way that it does not suffer from the above-mentioned issues and would be possible to tie the algorithm with a state estimation method resulting from the first goal. Specifically, these modifications will be made:

- combining local and global parameters to lower the dimension of the local models and validity functions,
- allowing separate dimension orders for the local models and the validity functions,
- improving the adaptation convergence stability while applying the algorithm incrementally,
- reducing the user defined parameters,
- allowing the interconnection if the algorithm with the state estimation methods according to the results of the first goal.

The output of this goal will be a modified adaptive RFWR algorithm for modelling nonlinear dynamic systems in the form of a Matlab library.

Hybrid method for the simultaneous modelling and state estimation of nonlinear dynamic systems

Following up on the previous goals, the third goal of this research is to further develop and implement the results from the first goal into the modified RFWR algorithm created in the second goal, resulting in a new hybrid method for both the modelling and state estimation of nonlinear dynamic systems. The main task is to achieve a reduction in the number of parameters a user needs to set and tune while simultaneously maintaining or even improving the overall performance of both the modelling and state estimation.

Again, the work will result in a Matlab library.

Case study on a real system

The algorithms developed in the second and third goals will need to be tested, first in a simulation and on a real system afterwards, which might reveal some possible shortcomings that may remain hidden when using simulated data (e.g., due to non-white noise or unexpected types of nonlinearity). Both algorithms will be tested throughout the development process on a suitable system (e.g., a rotary inverse pendulum, a magnetic ball levitation system or an automotive actuator, such as a throttle valve or an exhaust gas recirculation (EGR) valve) to ensure that the results are applicable in the real world.

These experiments will serve both as proof that the algorithms are working as intended and as an example implementation for a possible new user who intends to use the results in their own application or research.

4 Kalman filter with uncertain parameters

Based on our experience and previous research, as well as the current scientific literature [20, 5, 21, 22, 23, 24], it is safe to say that of the two covariance matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ representing the process and measurement noise, respectively, the process noise is the bigger issue. The measurement noise value can usually be estimated, measured, or acquired from the equipment datasheet. For this reason, and for the fact that only the ratio between $\boldsymbol{Q}$ and $\boldsymbol{R}$ matters, we will consider $\boldsymbol{R}$ to be known (at least in its order) throughout the rest of this thesis and focus on the methods on how to deal with $\boldsymbol{Q}$.

Before we discuss how to set $\boldsymbol{Q}$ correctly, we need to address various sources of imperfection and uncertainty that might cause it in the first place. We were able to identify several categories of effects that play a dominant role in practical applications.

The process noise can be caused by:

- a stochastic or an immeasurable input to the system
- a discretisation (or numerical integration) error
- a prediction model imperfection
  - wrong, incomplete, or imprecise model structure
  - inaccurate or uncertain parameters

The first item, a stochastic or an immeasurable input to the system, is the most studied case. We can find several algorithms for tuning the $\boldsymbol{Q}$ matrix, for example, in [5]. There is usually nothing that can be done about this issue to diminish its influence, as it is mostly part of the application. Examples are wind blowing, human behaviour, Brownian motion, or quantum effects. In our case, we can consider this issue negligible as we try to model or measure every effect influencing the system that we try to track.

The effect of discretisation can be serious and is often disregarded, however, with most dynamic systems in mechatronics, it can be diminished by using higher order discretisation methods or shorter sampling periods. Again, in our case, this effect can be considered negligible if treated carefully.

The last item on the list is the prediction model imperfection, which can be further divided into two categories. The first category covers situations where the model structure simply cannot represent the system it is supposed to model well enough. For example, when a linear model is applied on a nonlinear system or when special kinds of nonlinearities, such as friction effects, are not considered in the model structure. This is an issue that is very hard to deal with, and setting the $\boldsymbol{Q}$ matrix correctly is left as an ad hoc implementation. However, in the case of our research where we decided to use local linear approximation methods, it is safe to assume that this effect becomes negligible once the approximation converges.

That leaves us with the second category of the last item - inaccurate or uncertain model parameters. This issue is often regarded as the same problem as the imperfect model structure and receives very little scientific attention. We were able to locate only a single paper [25] dealing with a similar problem, however, especially in the case of local models, this issue becomes the dominant effect causing the process noise. We also believe that this issue is the dominant effect far more often than not, except when stochastic or immeasurable effects are present. Furthermore, as opposed to most of the other issues, we will show, in the following sections, that the parameter uncertainty can be dealt with and offers a unique way to set the $\boldsymbol{Q}$ matrix accordingly.

In ideal conditions, with all the KF requirements met, Figure 4.1 shows the KF performance based on different $\boldsymbol{Q}$ setting.

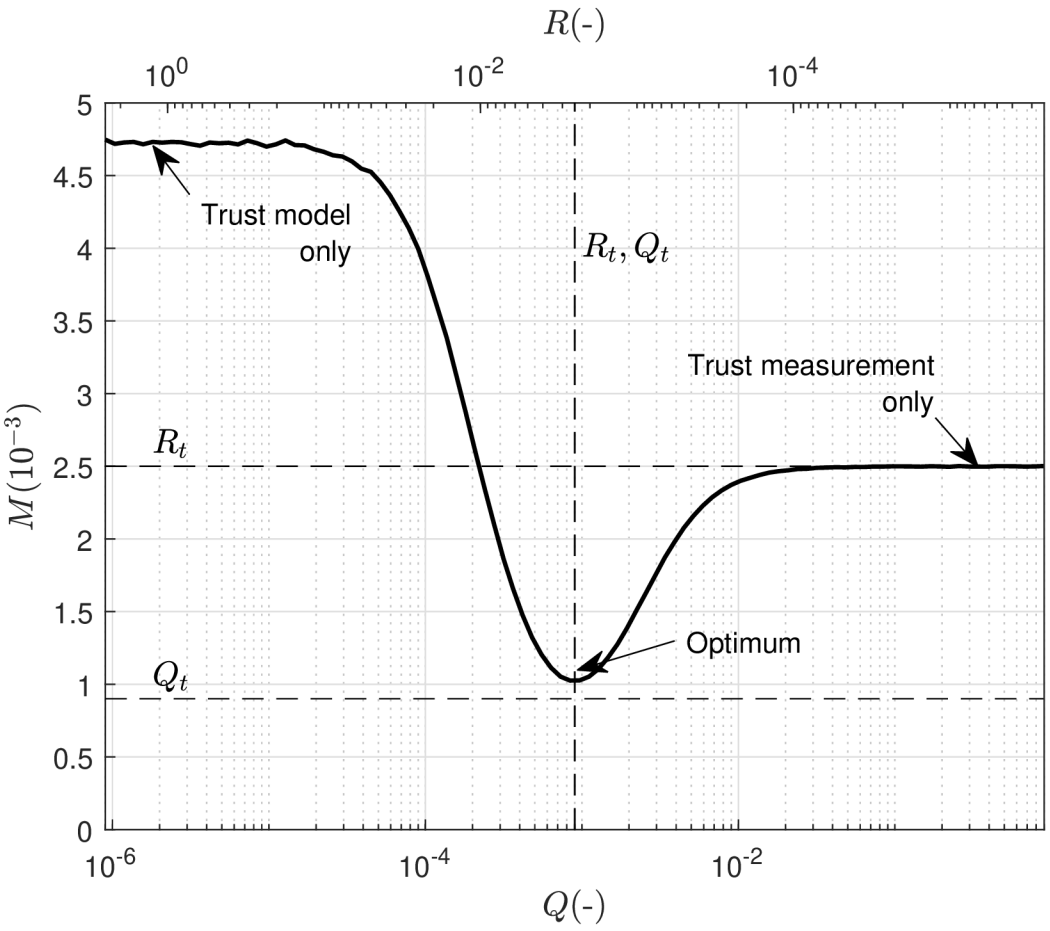

Figure 4.1: The Kalman filter performance measured as an MSE as a function of the $\boldsymbol{Q}$ and $\boldsymbol{R}$ values chosen along the axis of symmetry. The figure depicts the comparison of the performance metric to the true noise variance values and shows both the extreme and optimal cases of various $\boldsymbol{Q}$ and $\boldsymbol{R}$ settings.

4.1 Reformulating KF for uncertain parameters

In the previous section we summarised the effects causing the process noise and determined that the parameter uncertainty is the dominant effect that we should deal with. Similarly, as the Kalman filter algorithm deals with imprecisions in a stochastic way, we will consider the parameter uncertainties stochastic in the same Gaussian way. The reason for this assumption is that most parameter estimation algorithms use the Least squares method or one of its variants to determine the parameters, and this allows for the estimation of the uncertainty. We will expand on this idea in the following section.

First, consider an autonomous one-dimensional discrete system (4.1).

$$x_{k+1} = bx_k \tag{4.1}$$

where b is the only system parameter replacing the state matrix F, meaning, in this case, $F = b$.

The Kalman filter assumes the prediction model in the form (4.2), where w_k is the process noise, but it can also be seen as the complement to the model to make it fit perfectly.

$$x_{k+1} = bx_k + w_k \tag{4.2}$$

Assume that we use an estimated parameter $\hat{b}_k$ with an uncertainty represented by its variance $S_k = var(\hat{b}_k)$, meaning that the true value of $b \sim \mathcal{N}(S_k, \hat{b}_k)$. We can reformulate the model (4.1) into (4.3)

$$x_{k+1} = (\hat{b}_k + s_k)x_k \tag{4.3}$$

with $s_k \sim \mathcal{N}(S_k, 0)$ being the parameter noise in the same sense as the process noise w_k, meaning it is the complement to the perfect value. With this in mind, we can further expand the model according to (4.4).

$$\begin{aligned} x_{k+1} &= (\hat{b} + s_k)x_k \\ &= \hat{b}x_k + s_k x_k \\ &= \hat{b}x_k + w_k \end{aligned} \tag{4.4}$$

where $w_k = s_k x_k$.

This model corresponds to the original KF model (4.2) with the additional fact that the process noise w_k is now a function of the state x_k. This is an important and also expected fact when we consider the uncertainty of the parameters as the dominant source of the process noise.

The process noise covariance is used in the regular Kalman filter algorithm to predict the value of the state estimate covariance according to (2.23), assuming $w_k \sim \mathcal{N}(\boldsymbol{Q}_k, 0) \implies \boldsymbol{Q}_k = var(w_k)$. From this, we can expand the process noise covariance

matrix $\boldsymbol{Q}_k$ as (4.5).

$$\begin{aligned} \boldsymbol{Q}_k &= var(w_k) \\ &= var(s_k x_k) \\ &= \boldsymbol{S}_k x_k^2 \end{aligned} \tag{4.5}$$

Using this expansion, we can modify the state estimate prediction step (2.23) as (4.6). This specific expansion only works for one-dimensional systems, we will deal with its expansion to higher dimensions in a later section, after experimentally verifying the modification.

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^T + \boldsymbol{S}_k x_k^2 \tag{4.6}$$

The estimate covariance prediction step in the KF basically expands the covariance by an amount attributed to the process noise, thus keeping track with the state prediction step. First of all, (4.6) assumes that all the process noise comes from the parameter uncertainty and is covered by the term $\boldsymbol{S}_k x_k^2$, which depends on the actual state x_k as opposed to the regular term $\boldsymbol{Q}_k$.

The results show that with the correct setting, the modified state-dependent method for the KF estimate covariance prediction according to (4.6) performs comparably or even marginally better when set correctly as opposed to the original prediction method (2.23) in situations where the parameter uncertainty is the dominant cause of the process noise. The question that remains is how to set $\boldsymbol{S}$ correctly.

4.2 Setting the process noise covariance

In this section, we try to find the answer to the question of finding or estimating the optimal setting for the process noise covariance $\boldsymbol{S}$ when using the modified prediction method (4.6) introduced above.

In Section 4.1 we made the assumption that the imprecise estimation or stochastic uncertainty of the model parameters can be modelled as white Gaussian noise variance $\boldsymbol{S}$. Obviously, this is a simplification to a certain degree and this kind of assumption cannot always be met, however, there are several reasons to make it. First of all, it makes it possible to work with the uncertainty represented by a single number (or a matrix in a multidimensional case) and since it is the same assumption the Kalman filter framework makes (and guarantees optimality for), it makes it compatible with the rest of the KF algorithm. Second, often while estimating the model parameters, we also acquire a measure of how well the model fits the data, which may lead to an estimate of the correct setting for the $\boldsymbol{S}$ value. Especially in our case, when we chose to use the local linear approximation using LS and RLS for the parameter estimation, the model comes directly with the parameter uncertainty estimate. For the least squares method (LS), the uncertainty covariance estimate is known to be calculated according to (4.7),

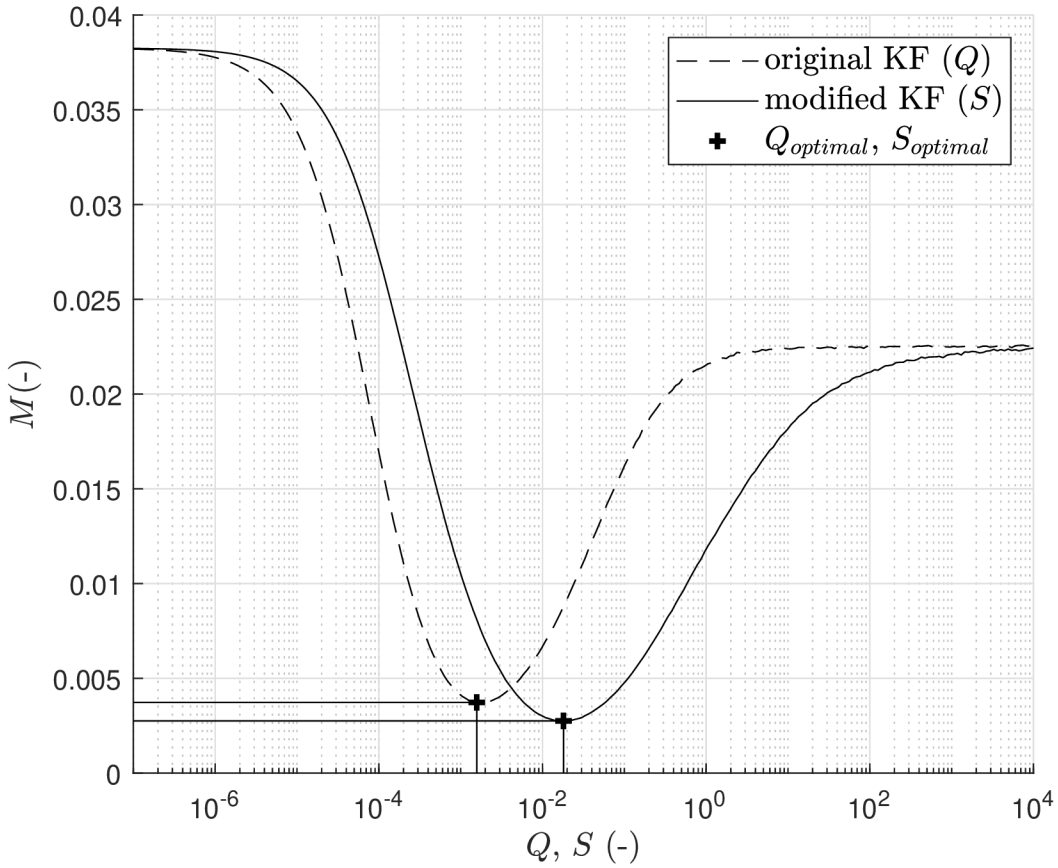

Figure 4.2: The simulation compares the effect of the original ($\boldsymbol{Q}$) and modified ($\boldsymbol{S}$) KF estimate covariance prediction step applied individually. Each point in the simulation represents an average of 25 simulations with the randomly picked parameter b with a 10% standard deviation. The results show that the modified version outperforms the original KF algorithm by about 35% in this specific scenario with uncertain parameters, with $\boldsymbol{S}_{optimal} = 2.75 \cdot 10^{-}3$ and $\boldsymbol{Q}_{optimal} = 3.73 \cdot 10^{-}3$.

$$var(\hat{b}) \approx \sigma^2 \left(X^T X\right)^{-1} \approx e^2 \left(X^T X\right)^{-1} \tag{4.7}$$

where σ^2 is the estimate error variance, which is theoretically unknown, hence, it is replaced by the estimate residual variance e^2 and $X^T X$ is the so-called cofactor matrix based on the data matrix X used for the LS estimate. See [26, 27, 28] for further details on the topic. Also, this parameter variance estimate corresponds to the P matrix in the RLS algorithm described.

Furthermore, we introduced an empirical formula which proved to be a very good estimate of the value of $\boldsymbol{S}$. This formula (4.8) is derived from the assumption that $\boldsymbol{S}$ must depend on the long-term variance w.r.t. zero of the signal we are tracking.

$$\hat{\boldsymbol{S}} = \frac{1}{2}\boldsymbol{B}E\left[x^2\right]^{-1} \tag{4.8}$$

Starting from the same simulation as in Figure 4.2, we estimate the value of $\boldsymbol{S}$ using

the empirical formula (4.8). We can see that the estimate is very close to the optimal value, at least in terms of the resulting filter performance.

The experiment, shown in Figure 4.3 expands on the previous one by studying only the effect of the parameter noise s in more detail, providing not only the average values but also the 95% confidence intervals for both the optimal and the estimated $\boldsymbol{S}$ values.

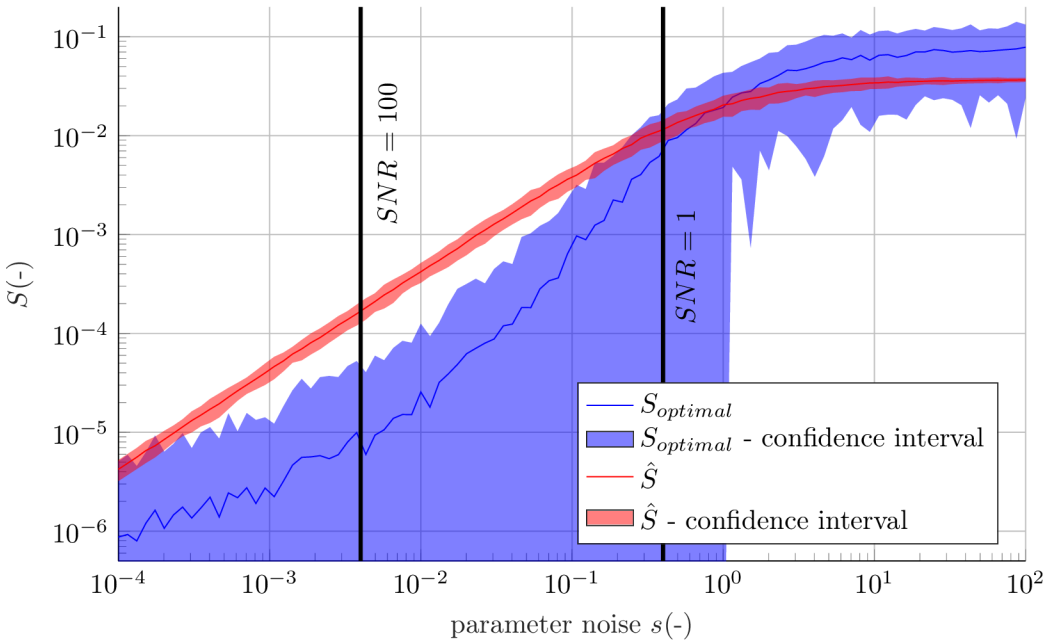

Figure 4.3: A simulation comparing the estimate and the optimal value of $\boldsymbol{S}$ for the modified Kalman filter with varying parameter noise s. Each point represents the average value of 100 independent simulations and its 95% confidence interval. The optimal value $\boldsymbol{S}_{optimal}$ was found using a gradient search optimisation method and $\boldsymbol{R}$ was set optimally in each simulation.

This experiment demonstrates that the $\boldsymbol{S}$ value estimate slightly overshoots the true optimal values in the interval with a reasonable SNR. The overshoot, at 0-2 orders of magnitude should still provide good enough performance only slightly under-performing the optimal values while keeping on the more stable side, providing more confidence in the measurement relative to the process prediction.

5 RFWR modification

In the previous chapter, we developed and experimentally verified a modified version of the Kalman filter suitable for hybrid dual estimation with local linear models. Before applying this modification at scale, in this chapter, we return to the original local modelling algorithm that we chose as our starting point, the Receptive Field Weighted Regression, to further develop and modify this method to be more suitable for use in the situations that we are dealing with in this thesis in the form of a user library.
[1]

The main functionality of the library is accessible to the user through just three methods.

There is the *learn* function, which takes both $\boldsymbol{x}$ and y as input, which have the meaning of a single corresponding data sample ($\boldsymbol{x}$, y) used to update the set of local models stored inside the object. The method has no output. This method implements the core functionality of the RFWR algorithm as described in [16], namely adding new and pruning obsolete local models, and updating the model parameters and their respective receptive fields. All the functions called inside the *learn* method are privately accessed methods of the *rfdelib* class. Figure 5.1 shows a simplified implementation of the learn method.

```
function    learn    (h,x,y)
    W = h.get_lm_weight(x,0,0); % calculate weight for every lm

    h.update_lm(x,y,W); % update lms using RLS and distance metric

    h.add_lm(x,y,W); % add new lm & rf if necessary

    h.prune_lm(W)% prune obsolete lm & rf if necessary
end
```

Figure 5.1: Implementation of the basic RFWR functionality in the *learn* method.

Lastly, there is the method *get_estimate*, which is used to calculate the actual y estimate corresponding to the input query point $\boldsymbol{x}$. The method also returns the size of the 95% confidence interval corresponding to the estimate.

It is actually quite simple to use the library as is shown in the pseudocode presented in Figure 5.2.

For user convenience, there are also publicly accessible methods which generate data for various possible visualisations describing the set of local models and the current approximation result, especially in lower-dimensional cases.

Compared to the original algorithm, the RFWR method was modified to be used in-

[1] In this chapter, we assume the reader is familiar with the basic concepts of (object-oriented) programming and Matlab programming language syntax.

```
LM = rfdelib(2); % initialise the LM object for two inputs

for i = % each datapoint index
    x(:,i) = get_system_input(); % measure or load input data
    y(i) = get_system_output(); % measure or load output data
    LM.learn(x(:,i),y(i))
end

xq = [0 0]; % the query point - the point of interest
yq = LM.get_estimate(xq); % calculate output estimate
```

Figure 5.2: Simple example of the library usage.

crementally in an online setting. The RLS learning algorithm is incremental by default, however, the approach for optimisation of the distribution of the receptive fields was supposed to be used on batch basis. To address this issue, we developed different optimisation strategies.

That being said, the original RFWR algorithm can be implemented in an incremental way, however, it turned out to be very hard to set the parameters correctly as the incremental model parameter estimation and receptive field distribution optimisation have low stability, especially with uneven data distribution in the input space. For example, with an incorrect parameter setting and a high data sampling rate, individual models would adjust to data points collected at one location much smaller than the local model validity region defined by the receptive field and would completely neglect past data gathered at different locations, sort of overfitting.

To address this issue, we added a datapoint buffer for every local model which is used to update the model parameter every time a new datapoint is collected, while the datapoints in the buffer are replaced by new ones on a random basis.

Another major modification which has the potential to greatly simplify the use of the library is the introduction of the option to set apart the dimensions of the local models and their respective receptive fields. In effect, this causes the local model to distribute receptive fields along only a subset of the input dimensions. This may be useful in situations where we expect the nonlinearity in the approximated function to be dependent only on some of the input quantities and be linear in others. This greatly improves the performance and stability, especially when there is only one nonlinear dimension. Typically, with mechanical systems, we expect the system to be linear in velocity while being nonlinear in position, also with multi-domain systems, typically only some of them contain nonlinearities.

The third significant add-on is the generalisation of the model input vector. In the original RFWR algorithm, there is the assumption that the local models generate and estimate a parameter for each of the system inputs plus a bias parameter, which is implemented through expanding the input vector by a unit constant. The library treats the data input vector and the model input vector as two separate things, linked by a private method called *lm_input_shuffler*. By default, this method acts the same way as the original algorithm - expands the input vector by the number one to generate the bias parameter, however, it allows for much more customisation. Generally, the local models used inside the RFWR framework do not have to be only linear combinations of input quantities, it can take the form of any function of the inputs that is linear in parameters.

One separate modification to the original RFWR algorithm that we made is the addition of several different methods for optimising the distribution of the receptive fields.

Naturally, the first method that the *rfdelib* class implements is the original algorithm described in [16]. This is the only part of the library taken from another source (the authors' implementation published in [29]) and is only modified to fit the library framework.

The second (*Heuristic*) method uses the analytical Jacobian matrix of the weight function to be able to correctly adjust the elements of the distance inducing matrix M, however, it uses a simple heuristic decision rule to choose if the region of validity of the given local model should be made larger, smaller, or stay the same with respect to the actual datapoint and the local model's long-term performance. This method was first introduced in [15] and later improved.

The third (*Random*) update method that the library implements is based on the stochastic optimisation approach. It only works with diagonal distance inducing matrices which, on the one hand, do not allow for such precise receptive field optimisation and usually require a few more local models to cover the same input space, however, on the other hand, it requires much less computational power and brings more stability. In effect, the RF basis functions only scale along the RF input dimensions.

The last (*Numerical*) update algorithm that the *rfdelib* class implements is a method using the numerical approximation of the criterion function $\boldsymbol{J}$ gradient. In this case, we work directly with the true criterion gradient $\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{M}}$ instead of the simplified $\frac{\partial w}{\partial \boldsymbol{M}}$.

Further, the library implement various minor improvements and modifications such as different initialisation methods for newly added RFs, using the weighted version of the RLS algorithm and a number of ease-of-use methods to get set save, load or plot all the necessary data describing the set of local models

To test the behaviour of the *rfdelib* library, we use a second-order nonlinear system, which might represent a mechanical oscillator with a nonlinear spring. The system (5.1) contains a nonlinearity that spans only a part of the state space of the system, to force the algorithm to deal with nonlinearities on different scales.

$$m\ddot{x} = b\dot{x} + k_1 x + k_3 x^3 + k_5 x^5 + k_6 e^{-\frac{(x-k_7)^2}{k_8}} + u \tag{5.1}$$

Figure 5.3 shows the situation with 6000 datapoints. It is clear that the distribution of the receptive fields is optimised, meaning there are fewer, larger receptive fields in regions with less significant nonlinearities and the local models begin to cluster more around $x \approx -0.6$ to fit the highly nonlinear part of the function. Also, the confidence interval overall is much narrower.

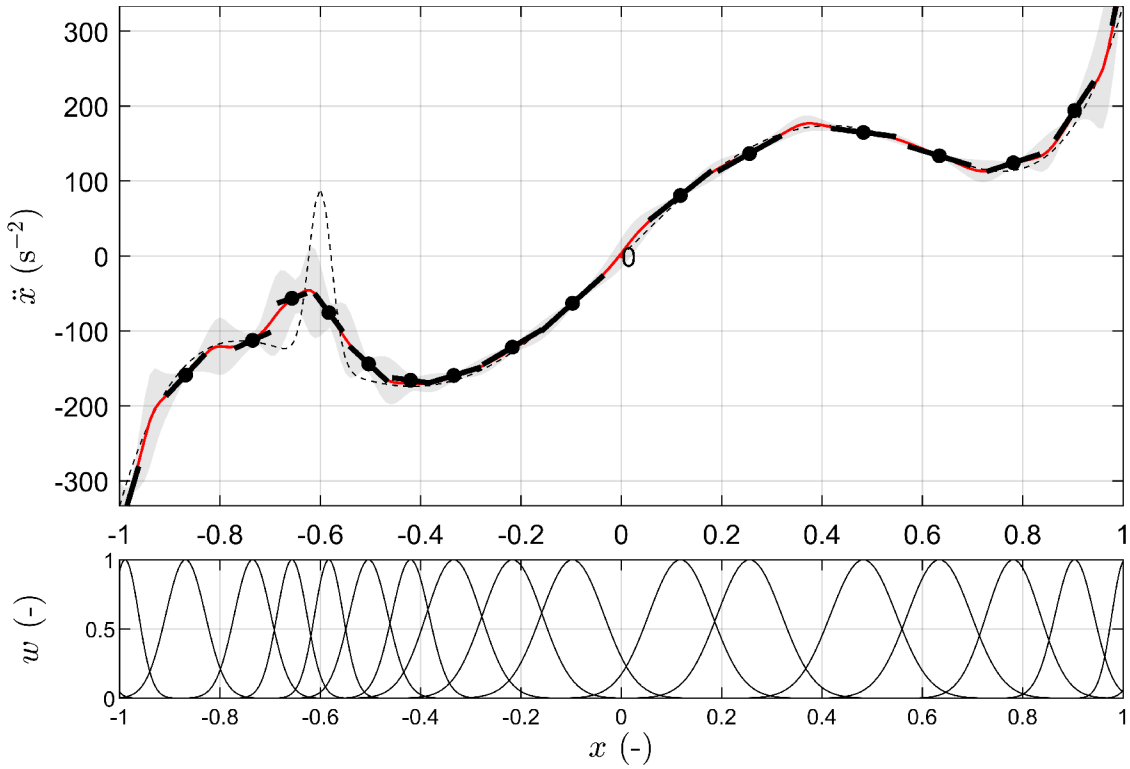

Figure 5.3: *rfdelib* approximation of the nonlinear system dynamics with 6000 datapoints.

6 Dual estimation based on RFWR and Kalman filter

In this chapter, we describe the final extension of the *rfdelib* library introduced in the Chapter 5. The extension implements the ideas on the Kalman filter modification presented in Chapter 4. In terms of the various simultaneous estimation approaches described in Section 2.3, the combination of the RFWR algorithm for model and parameter estimation and the modified KF for state estimation forms a hybrid dual estimation algorithm, hence, the abbreviation RFDE in *rfdelib* that stands for Receptive Field Dual Estimation.

The *rfdelib* library implements the hybrid dual estimation algorithm separately for each existing local model in the RFWR structure. Figure 6.1 presents a diagram for the RFDE algorithm, showing the interconnection between the RFWR local model set and the modified KF calculated for each of the local models.

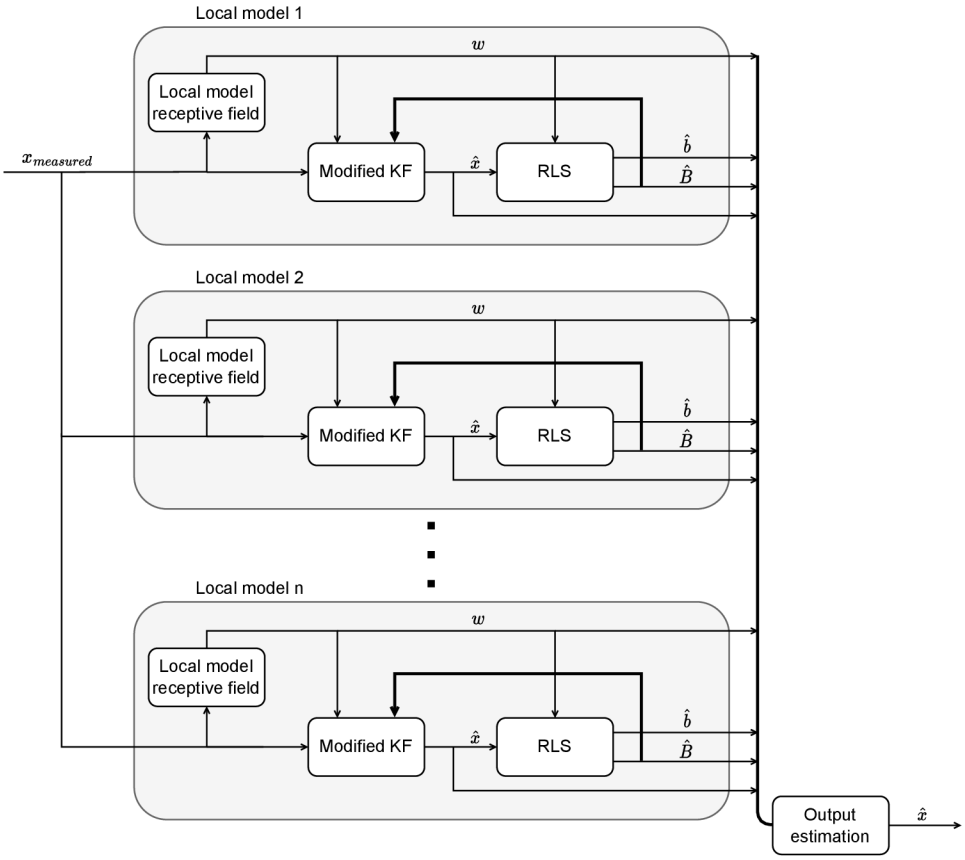

Figure 6.1: The diagram of the *RFDE* algorithm showing the interconnection between the RFWR local model set and the modified KF calculated for each of the local models. $\boldsymbol{x}$ represents the system state, w represents the actual local model weight, $\boldsymbol{b}$ represents the local model parameters and $\boldsymbol{B}$ represents the model parameter covariance.

To test the *rfde* library under real conditions, we performed a case study of hybrid simultaneous estimation on a laboratory model of a magnetic manipulator, the same one that was used in [30] and initially presented in [31].

The magnetic manipulator, shown in Figure 6.2, consists of a row of four coils and an iron ball. The position of the ball, measured using a laser distance sensor can be controlled by varying currents in the coils, which are driven by a specialised power electronic unit set through Matlab commands. The communication between the Matlab environment and the power control unit is carried out using the Humusoft MF634 PCIe IO card [32].

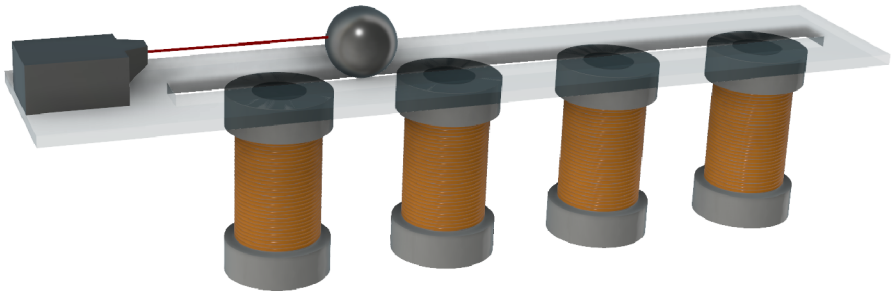

Figure 6.2: 3D render of the magnetic manipulator with a row of coils and an iron ball in a linear pathway. Taken from [31] and [30], respectively.

When studying the dynamics of the system, it is important to create a model for a single coil, as the effect of multiple coils can be viewed as additive ([31, 30]), especially when the system is intended to be controlled in such a way that only a single coil is active at any time. For this reason, we focus on the simultaneous estimation of a system with the iron ball and a singe coil. This system can generally be described using an explicit state model (6.1).

$$\ddot{x} = f\left(x, \dot{x}, u\right) \tag{6.1}$$

where x is the position of the ball and u is the system input corresponding to the coil current.

First, we set the RFDE algorithm so that only the position of the ball is considered when plating the receptive fields of the local models, making the distribution one-dimensional. The result is shown in Figure 6.3 after learning with 7400 datapoints. We can see that the algorithm was able to adapt and to find the shape of the underlying nonlinear function similar, however, it contains a great deal of uncertainty and the shape of the function is not precise.

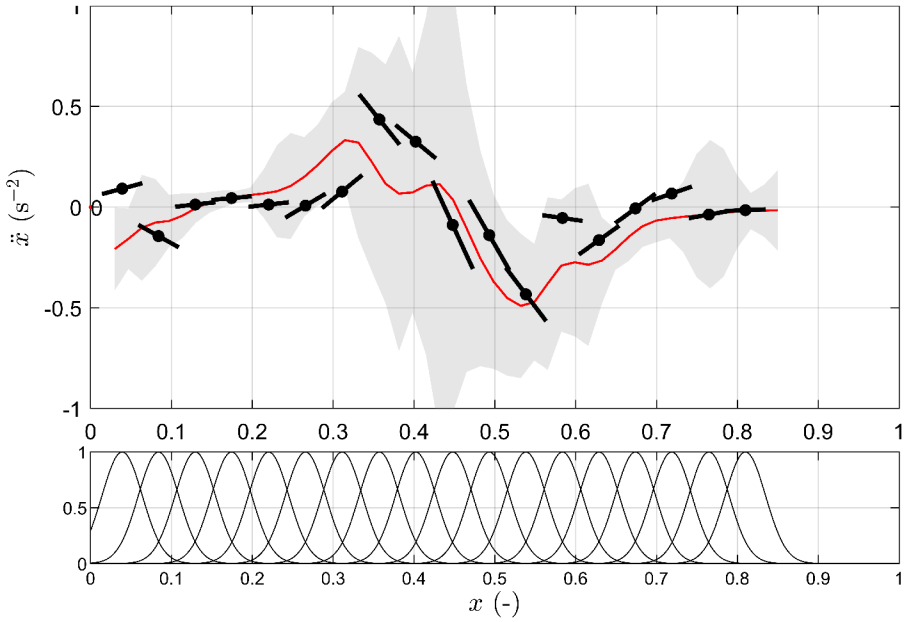

Figure 6.3: The model shape learned by the *RFDE* algorithm with one-dimensional RF distribution along the x dimension after 7400 datapoints. The figure shows the dynamic function shape for $u = 0.5$ A and also the distribution of the RFs.

In the second experiment, we set the RFDE algorithm so that both the ball's position and the velocity are considered when placing the receptive fields. The result is shown in Figure 6.4. The figure shows the shape of the dynamics of the system in the RF space for $u = 0.5$, as well as the size of the model confidence interval, after 17400 datapoints. Again, we can see that the characteristic nonlinear shape is present along the x axis, however, the coil effects are lower in higher velocities, representing the effects of the eddy currents.

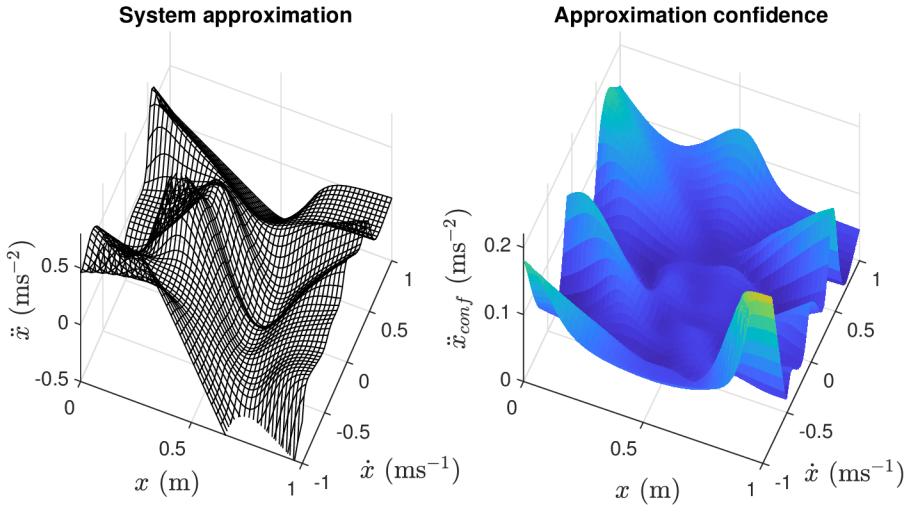

Figure 6.4: The model shape learned by the *RFDE* algorithm with two-dimensional RF distribution along the x and $\dot{x}$ dimensions after 17400 datapoints. The figure shows the dynamic function shape for $u = 0.5$ A and the corresponding size of the confidence interval.

7 Conclusion

The research described in this thesis was motivated by engineering experience and previous research, published in [33], concerning all three basic tasks described in the Introduction, that is, the modelling, state estimation, and control algorithm design for nonlinear dynamic systems. It turns out that, individually, these tasks are theoretically mostly solved and that the used methods work. However, their practical implementation raises many questions for which there are no clear answers. Also, while solving these tasks separately, we often arrive at suboptimal solutions due to the criteria used for the separate model parameter estimation, and the observer or controller tuning is defined as being unrelated even though they all contribute to the overall control quality.

The four main goals for this dissertation were defined in the presented thesis and approved at the State Doctoral Exam:

1. Research of the mutual relationships in typical algorithms used for estimating the states and parameters of dynamic system models.

2. Modification of the RFWR method for its use with typical mechatronic systems.

3. Developping a Hybrid method for the simultaneous modelling and state estimation of nonlinear dynamic systems.

4. Implementation and testing of the resulting algorithms in a real system.

Within the first goal, we mainly studied the uncertainty of dynamic system parameters estimated on the basis of noisy data. This uncertainty is one of the sources of estimation errors in state estimation algorithms. We identified it as being dominant in situations where the structure (ordinary differential equation) of the model can be reliably determined, we do not work with principally stochastic systems, and the discretisation error is also not significant due to the sampling period being orders of magnitude shorter than the natural time constants of the system in question. Chapter 4 further describes the use of our discoveries for the reformulation of the classical Kalman filter algorithm for this very specific situation. We were able to achieve recognisable improvement in the state estimation accuracy, but most importantly, we were able to find an empirical approach for determining the tuning parameter most critical for the Kalman filter's practical implementation - the process noise covariance. Thanks to this development, we were able to create a parameterless link between the modified Kalman filter and a method for the parameter estimation of a local linear model of an otherwise nonlinear system. This link forms a hybrid dual estimation approach. We were also able to demonstrate its functionality and improved state and parameter estimation accuracy compared to that of a typical approach, thus fulfilling the first goal of the thesis.

Chapter 5 deals with the modification of the Receptive Field Weighted Regression (RFWR) originally described in [16, 18] and the development of a user library for MATLAB. We have rewritten the method into an incremental version to be used online without

the need for batch data processing. This modification was motivated by the inherent potential of incremental methods to work as adaptive approximators in situations in which the system that we observe can change its behaviour. For example, due to the wear and tear of its parts, changing external conditions, etc.

Additionally, we expanded the modified RFWR algorithm by several useful elements. Most importantly, allowing for a separate number of dimensions in the local models and the corresponding receptive fields. This is useful in situations where the nonlinearity of the system is exclusively or dominantly dependent on a subset of the system states and when other system dynamics are presumed to be linear. Other modifications included minor adjustments, such as four different variants of the receptive field optimisation algorithm, an improved computational performance, a learning data buffer for each local model, calculation of the resulting approximation confidence interval, etc. Thus, the second goal can also be considered achieved.

Furthermore, in Chapter 6, we describe the *Hybrid Method for the Simultaneous Modelling and State Estimation of Nonlinear Dynamic Systems* built upon the results of previous research goals. This method, in the form of dual estimation, combines the RFWR and KF modified for the situation with uncertain parameters. We tested the hybrid dual estimation approach through several simulation experiments and finally on data measured in a real system of a magnetic manipulator, achieving the third and fourth research goal of the thesis.

7.1 Thesis achievements

- **Categorisation of error sources for the Kalman filter (KF) process model**
  In Chapter 4, we described and categorised the possible sources of errors that influence the KF process model, which may arise in common engineering applications. This methodology can be helpful in practical KF implementations and especially in tuning the values of the process noise covariance.

- **Modification of the KF for situations with inaccurate and uncertain process model parameters**
  Based on previous research, published in [34], we modified a part of the Kalman filter algorithm, specifically the state vector covariance prediction step, to better suit the situation where the dominant source of the error is inaccurate or contains uncertain process model parameters, especially parameters estimated based on noisy data. In our experience, this situation is far more common than is assumed, although very little scientific attention is paid to it. The results show that this modified KF gives better results in conditions with the parameter uncertainty being the dominant source of the process noise.

- **Empirical approach for setting the process noise covariance**
  We found and experimentally verified an empiric formula to find the value of the process noise covariance for the Kalman filter in this very specific situation, greatly simplifying the practical implementations. It turned out that it is mostly important to set the correct order of the process noise covariance values and this empirical formula proved to give an a reasonably accurate and robust estimate.

- **Linking the KF and RLS algorithms for the dual estimation over a local linear model**
  We further modified the Kalman filter algorithm so that it can be used with local linear models that are linked to a parameter estimation method, for example, RLS. Mainly, this means implementing the KF in a way that it can work in a shifted state and input space corresponding to the centre of the local model validity function and using the validity function to weight the RLS estimate as well as the KF prediction.
- **Modification of the RFWR algorithm**
  We modified the RFWR function approximation method, which is based on the principle of local linear models described in [16], so that it is better suited to be used with common low-order mechatronic systems, which typically have a significant nonlinearity dependence on a subset of states, and the state space is formed by sequences of quantities in integral-derivative relationships. Some of these modifications were published in [30]. We also developed a user library for the MATLAB language.
- **Hybrid dual estimation**
  By combining the modified RFWR method and the modified Kalman filter, we developed a *Hybrid Method for the Simultaneous Modelling and State Estimation of Nonlinear Dynamic Systems*, which uses a separate Kalman filter for each local model whose parameters are estimated. We validated the functionality of this method both in simulation and experimentally, and further developed a user library for the MATLAB language, which allows for its simple and fast practical implementation.

References

[1] BOHNER, Martin; PETERSON, Allan. Dynamic Equations on Time Scales. *Dynamic Equations on Time Scales*. 2001. Available from DOI: 10.1007/978-1-4612-0201-1.

[2] ELLNER, Stephen P.; GUCKENHEIMER, John. Dynamic models in biology. *Dynamic Models in Biology*. 2011, vol. 9781400840960, pp. 1–329. ISBN 9781400840960. Available from DOI: 10.2307/j.ctvcm4h1q.

[3] NELLES, Oliver. Nonlinear System Identification. *Nonlinear System Identification*. 2001. Available from DOI: 10.1007/978-3-662-04323-3.

[4] KALMAN, R. E. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*. 1960, vol. 82, no. 1, pp. 35–45. ISSN 0021-9223. Available from DOI: 10.1115/1.3662552.

[5] *Kalman and Bayesian Filters in Python by rlabbe*. [N.d.]. Available also from: http://rlabbe.github.io/Kalman-and-Bayesian-Filters-in-Python/.

[6] SALAU, Nina P.G.; TRIERWEILER, Jorge O.; SECCHI, Argimiro R.; MARQUARDT, Wolfgang. A New Process Noise Covariance Matrix Tuning Algorithm for Kalman Based State Estimators. *IFAC Proceedings Volumes*. 2009, vol. 42, no. 11, pp. 572–577. ISBN 9783902661548. ISSN 1474-6670. Available from DOI: 10.3182/20090712-4-TR-2008.00092.

[7] PONCELA, Marta; PONCELA, Pilar; PERÁN, José Ramón. Automatic tuning of Kalman filters by maximum likelihood methods for wind energy forecasting. *Applied Energy*. 2013, vol. 108, pp. 349–362. ISSN 0306-2619. Available from DOI: 10.1016/J.APENERGY.2013.03.041.

[8] CHEN, Zhaozhong; HECKMAN, Christoffer; JULIER, Simon; AHMED, Nisar. Weak in the NEES?: Auto-Tuning Kalman Filters with Bayesian Optimization. *2018 21st International Conference on Information Fusion, FUSION 2018*. 2018, pp. 1072–1079. ISBN 9780996452762. Available from DOI: 10.23919/ICIF.2018.8454982.

[9] MU, Tingting; NANDI, Asoke K. Automatic tuning of L2-SVM parameters employing the extended Kalman filter. *Expert Systems*. 2009, vol. 26, no. 2, pp. 160–175. ISSN 1468-0394. Available from DOI: 10.1111/J.1468-0394.2009.00469.X.

[10] CAI, Levi; BOYACIOGLU, Burak; WEBSTER, Sarah E.; VAN UFFELEN, Lora; MORGANSEN, Kristi. Towards auto-tuning of kalman filters for underwater gliders based on consistency metrics. *OCEANS 2019 MTS/IEEE Seattle, OCEANS 2019*. 2019. ISBN 9780578576183. Available from DOI: 10.23919/OCEANS40490.2019.8962573.

[11] CAMPESTRINI, Christian; HEIL, Thomas; KOSCH, Stephan; JOSSEN, Andreas. A comparative study and review of different Kalman filters by applying an enhanced validation method. *Journal of Energy Storage*. 2016, vol. 8, pp. 142–159. ISSN 2352-152X. Available from DOI: 10.1016/J.EST.2016.10.004.

[12] PEI, Yan; BISWAS, Swarnendu; FUSSELL, Donald S.; PINGALI, Keshav. An Elementary Introduction to Kalman Filtering. *Communications of the ACM*. 2017, vol. 62, no. 11, pp. 122–133. ISSN 15577317. Available from DOI: 10.48550/arxiv.1710.04055.

[13] ASWAL, Neha; BHATTACHARYA, Baidurya; SEN, Subhamoy. Joint and Dual Estimation of States and Parameters with Extended and Unscented Kalman Filters. *Recent Developments in Structural Health Monitoring and Assessment — Opportunities and Challenges*. 2022, pp. 223–252. Available from DOI: 10.1142/9789811243011{_}0008.

[14] LJUNG, Lennart. System identification : theory for the user. 1999, p. 609. ISBN 978-0136566953.

[15] BRABLC, Martin. *Control Of Nonlinear Systems Using Local Approximation Methods*. Brno, 2016. PhD thesis. Brno University of Technology.

[16] SCHAAL, Stefan; ATKESON, Christopher G. Constructive Incremental Learning from Only Local Information. *Neural Comput.* 1998, vol. 10, no. 8, pp. 2047–2084. ISSN 0899-7667. Available from DOI: 10.1162/089976698300016963.

[17] VIJAYAKUMAR, S; D'SOUZA, A; SHIBATA, T; CONRADT, J; SCHAAL, Stefan. Statistical learning for humanoid robots. *AUTONOMOUS ROBOTS*. 2002, vol. 12, no. 1, pp. 55–69. Available from DOI: 10.1023/A:1013258808932.

[18] ATKESON, Christopher G.; MOORE, AW; SCHAAL, Stefan. Locally weighted learning. *ARTIFICIAL INTELLIGENCE REVIEW*. 1997, vol. 11, no. 1-5, pp. 11–73. Available from DOI: 10.1023/A:1006559212014.

[19] ATKESON, Christopher G.; MOORE, AW W; SCHAAL, Stefan. Locally weighted learning for control. *ARTIFICIAL INTELLIGENCE REVIEW*. 1997, vol. 11, no. 1-5, pp. 75–113. Available from DOI: 10.1023/A:1006511328852.

[20] CHUI, C. K.; CHEN, G. (Guanrong). Kalman filtering : with real-time applications. 2009, p. 229. ISBN 3642099661.

[21] R. ANANTHASAYANAM, Mudambi. Tuning of the Kalman Filter Using Constant Gains. *Introduction and Implementations of the Kalman Filter*. 2019. Available from DOI: 10.5772/INTECHOPEN.81795.

[22] HOUTEKAMER, Peter L.; MITCHELL, Herschel L.; DENG, Xingxiu. Model Error Representation in an Operational Ensemble Kalman Filter. *Monthly Weather Review*. 2009, vol. 137, no. 7, pp. 2126–2143. ISSN 1520-0493. Available from DOI: 10.1175/2008MWR2737.1.

[23] DRÉCOURT, Jean Philippe; MADSEN, Henrik; ROSBJERG, Dan. Bias aware Kalman filters: Comparison and improvements. *Advances in Water Resources*. 2006, vol. 29, no. 5, pp. 707–718. ISSN 0309-1708. Available from DOI: 10.1016/J.ADVWATRES.2005.07.006.

[24] FITZGERALD, Robert J. Divergence of the Kalman Filter. *IEEE Transactions on Automatic Control*. 1971, vol. 16, no. 6, pp. 736–747. ISSN 15582523. Available from DOI: 10.1109/TAC.1971.1099836.

[25] XIE, Lihua; DE SOUZA, Carlos E. Robust Kalman Filtering for Uncertain Discrete-Time Systems. *IEEE Transactions on Automatic Control*. 1994, vol. 39, no. 6, pp. 1310–1314. ISSN 15582523. Available from DOI: 10.1109/9.293203.

[26] FARAWAY, J.J. *Practical Regression and Anova Using R*. University of Bath, 2002. Available also from: https://cran.r-project.org/doc/contrib/Faraway-PRA.pdf.

[27] HAYASHI, Fumio. Econometrics. 2000, p. 683. ISBN 978-0691010182.

[28] VAN DE GEER, Sara A; EVERITT, Brian S; HOWELL, David C. Least Squares Estimation. *Encyclopedia of Statistics in Behavioral Science*. 2005, vol. 2, pp. 1041–1045. ISBN 978-0-470-86080-9. Available also from: https://stat.ethz.ch/~geer/bsa199_o.pdf.

[29] VIJAYAKUMAR, Sethu. *Software (Sethu Vijayakumar)*. [N.d.]. Available also from: https://homepages.inf.ed.ac.uk/svijayak/software/.

[30] BRABLC, Martin; ZEGKLITZ, Jaň; GREPL, Robert; BABUŠKA, Robert. Control of Magnetic Manipulator Using Reinforcement Learning Based on Incrementally Adapted Local Linear Models. 2021. Available from DOI: 10.1155/2021/6617309.

[31] DAMSTEEG, Jan Willem; NAGESHRAO, Subramanya P.; BABUŠKA, Robert. Model-based real-time control of a magnetic manipulator system. *2017 IEEE 56th Annual Conference on Decision and Control, CDC 2017*. 2018, vol. 2018-January, pp. 3277–3282. ISBN 9781509028733. Available from DOI: 10.1109/CDC.2017.8264140.

[32] *MF634 — Humusoft*. [N.d.]. Available also from: https://www.humusoft.cz/datacq/mf634/.

[33] SOVA, Václav. *Adaptivní řízení elektromechanických aktuátorů s využitím do- předného kompenzátoru založeného na více-modelovém přístupu*. Brno, 2018. PhD thesis. Brno University of Technology.

[34] NAJMAN, J.; BRABLC, M.; RAJCHL, M.; BASTL, M.; SPÁČIL, T.; APPEL, M. *Advances in Intelligent Systems and Computing*. Vol. 1044, Monte carlo based detection of parameter correlation in simulation models. 2020. ISBN 9783030299927. ISSN 21945365. Available from DOI: 10.1007/978-3-030-29993-4{_}7.

Curriculum Vitae

Personal Information:

Name:	Ing.Martin Brablc
Date of Birth:	25.12.1991
Citizenship:	CZ
Email:	martin.brablc@vutbr.cz

Education

2004 - 2011	Komenský Grammar School Havířov
2011 - 2016	Bc. and Ing., Mechatronics, FME Brno University of Technology
2016 - to date	Ph.D., Engineering Mechanics, FME Brno University of Technology

Employment

2011 - 2014	Piloun spol. s.r.o., Technician
2016	Norma Group CZ a.s., R&D intern
2017	CIIRC Prague, junior research scientist
2017 - to date	FME Brno University of Technology, assistant
2016 - to date	Mechsoft s.r.o., R&D engineer

Author's publication report as of January 2023

Type	**Count**	**Citations**
Scientific journal article Q1 (First quartil)	1	20
Scientific journal article Q2 (Second quartil)	3	2
Conference paper	9	16

Language competence

Czech	native
English	fluent (C2)
Spanish	basic (B1)

Technical competence & skills

Programming - C (embedded), Matlab, Python
signal processing, PCB design and testing, laboratory equipment, sensors

Abstract

This Doctoral thesis deals with the development of a new hybrid method for the dual estimation of states and parameters of nonlinear dynamic systems based on the idea of local linear models, which uses the estimation of the uncertainty of the model parameters to automatically adjust the parameters of the Kalman filter (KF), thus greatly simplifying its deployment and adjustment in practical applications. In the first part, the dissertation summarises the current state of knowledge in the field of dynamic systems, simultaneous estimation, KF and modelling of nonlinear dynamic systems. Then, in two separate chapters, it discusses the modification of KF for situations where inaccurate model parameters are the dominant influence causing process noise, and the modification of the *Receptive field weighted regression (RFWR)* method so that it can be used for dual estimation. Finally, the paper describes the developed hybrid method composed of modified RFWR and KF algorithms called *Receptive field dual estimation - (RFDE)* and demonstrates its performance on simulation and real data.

Abstrakt

Tato Disertační práce se zabývá vývojem nové hybridní metody pro současné odha- dování stavů a parametrů nelineárních dynamických systémů, založené na myšlence lokálních lineárních modelů, která využívá odhad nejistoty parametrů modelu pro automatické nastavení některých parametrů Kalmanova filtru (KF), čímž se výrazně zjednodušuje její nasazení a nastavení v praktických aplikacích. V první části se disertační práce věnuje shrnutí aktuálního stavu poznání v oblasti dynamických systémů, simultánní estimace, KF a modelování nelineárních dynamických systémů. Následně se ve dvou samostatných kapitolách věnuje modifikaci KF pro situace, kde dominantním vlivem způsobujícím procesní šum jsou nepřesné parametry modelu, a dále modifikaci metody *Receptive field weighted regression (RFWR)* tak, aby mohla být použita pro duální estimaci. Nakonec práce popisuje vyvinutou hybridní metodu složenou z modifikovaných algoritmů RFWR a KF nazvanou *Receptive field dual estimation - (RFDE)* a demonstruje její funkčnost na simulačních i reálných datech.