

Minimum Energy Estimation of Nonlinear Dynamical Systems in Continuous and Discrete Time

by

Albert Tres Vilanova

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Applied Mathematics

Waterloo, Ontario, Canada, 2025

© Albert Tres Vilanova 2025

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In many applications, accurately estimating a system's state is crucial for effective control. This thesis gives an in-depth introduction to the minimum energy estimator (MEE), an optimal estimator that is based on the minimum energy cost function. Although less widely used than the extended Kalman filter or the unscented Kalman filter, the MEE offers a rigorous optimality guarantee. Using tools from optimal control theory - such as the Pontryagin maximum principle and the Hamilton-Jacobi-Bellman (HJB) equation - we derive both continuous and discrete time formulations of the MEE. Conditions for the stability and converge of the MEE are reviewed. We also demonstrate that the MEE coincides with the Kalman filter in the case of linear dynamics. The derivation leads to an estimator equation for the MEE that requires solving an HJB equation, prompting the introduction of various numerical methods to address this challenge. To further explore the relationship between continuous and discrete time filtering, simulations are conducted to compare the accuracy of the discrete time Kalman filter and a "hybrid" Kalman filter, where the continuous time filter is discretized in the update step.

Acknowledgements

To begin with, I would like to express my sincere gratitude to my supervisor, Professor Kirsten Morris, for providing me with invaluable mentorship, support, and patience throughout my Master's program.

Moreover, I would like to extend my gratitude to my committee members for agreeing to read and review my thesis. I am also appreciative to the faculty members and staff for their support and contributions to my academic journey.

Finally, I am thankful to my family and friends for their unwavering support and understanding.

Dedication

This work is dedicated to my family and to my partner, thank you for your never-ending patience and support.

Table of Contents

Author's Declaration	ii
Abstract	iii
Acknowledgements	iv
Dedication	v
List of Figures	viii
1 Introduction	1
2 Overview of Optimal Control	3
2.1 The Basic Problem	3
2.2 The Cost Functional	4
2.3 A Brief Introduction to Calculus of Variations	5
2.4 Pontryagin Minimum Principle	6
2.5 Finite-Time Linear-Quadratic Regulator Problem Solved Using Pontryagin's Minimum Principle	9
2.6 Dynamic Programming and the Hamilton-Jacobi-Bellman Equation	13
2.6.1 Viscosity Solutions to Partial Differential Equations	17
2.7 Linear-Quadratic Regulator Problem Solved Using Dynamic Programming	19
2.8 Discussion of Pontryagin Minimum Principle and Hamilton-Jacobi-Bellman Equation	21
3 Optimal State Estimation	23
3.1 Optimal Estimation	25
3.2 Minimum Energy Estimation in Continuous Time	29

3.3	Derivation of the Continuous Time Minimum Energy Estimator Dynamics	33
3.4	Continuous Time Minimum Energy Estimation in the Linear Case	37
3.5	Convergence of the Continuous Time Minimum Energy Estimator	42
3.6	Minimum Energy Estimation in Discrete Time	49
3.7	Discrete Time Linear Case	53
3.8	Relating Discrete and Continuous-Time Filters	54
3.8.1	Forward Euler Discretization of the Continuous-Time Kalman Filter	54
3.9	Approximate Approaches to Estimation	56
3.9.1	Extended Kalman Filter	56
3.9.2	Krener’s Taylor Polynomial discrete-time MEE scheme	57
3.9.3	Unscented Kalman Filter	58
3.9.4	Particle Filter	59
4	Numerical Methods for Solutions to the HJB	60
4.1	Classical Methods	61
4.2	Neural Network Methods	63
5	Simulations	65
5.1	Discrete Time vs “Hybrid” Kalman Filter for Mass Spring System	65
5.2	Discrete Time vs “Hybrid” Kalman Filter for Multi Mass Spring System .	71
5.3	Discussion on Small or Large Noise Covariance R	74
6	Conclusions and Future Work	79
	References	80
	APPENDICES	89
	A Cost function derivation	90
	B Link to Code Repository	93

List of Figures

2.1	Driver Automobile system, replicated from figure 1.1 in [74].	9
2.2	Subsolution and supersolution at a point x_0 where ∇V is not defined. Figure 3.4 replicated from [94].	19
5.1	Mass Spring Position Estimates Comparison.	68
5.2	Mass Spring Position Estimates Errors.	69
5.3	Mass Spring Velocity Estimates Comparison.	70
5.4	Mass Spring Velocity Estimates Errors.	71
5.5	Multi Mass Spring Position Estimates and Errors.	72
5.6	Multi Mass Spring Velocity Estimates and Errors.	73
5.7	Position Estimate with Small (1) and Large (50) Factor for Noise Covariance.	74
5.8	Velocity Estimate with Small (1) and Large (50) Factor for Noise Covariance.	75
5.9	Root Mean Squared Errors for Factors 1 Through 50 for Position and Velocity Estimates.	76
5.10	Position Estimates for Mass Spring System for Three Different Factors Multiplying the Covariance R.	77
5.11	Velocity Estimates for Mass Spring System for Three Different Factors Multiplying the Covariance R.	78

Chapter 1

Introduction

Dynamical systems are used to model phenomena in a wide range of areas such as medicine, engineering, finance and more. Often, this modeling includes nonlinear dynamics to capture the complexity of certain systems. In certain applications, the goal of having a mathematical model isn't simply to observe and predict dynamics but to study the system with the goal of applying a control to it. Yet, controlling a system requires precise knowledge; a level of accuracy beyond that provided by the system dynamics or the noisy, incomplete measurements that may accompany it. In such cases, estimation is used to blend the model and the available measurements to obtain a more accurate picture of reality.

Consider the problem of spacecraft navigation. The spaceship's motion is influenced by gravitational forces from celestial bodies, thruster-induced perturbations, and other factors; all of which introduce nonlinearities and disturbances into its dynamics. At the same time, measurements from onboard sensors are subject to errors, bias, slow frequency or loss of signal, complicating the estimation process.

Precise knowledge of the spacecraft's state, including position, velocity, and altitude, is essential for course corrections and ensuring mission success. Estimation techniques such as the Kalman filter, extended Kalman filter or the minimum energy estimator are used in such cases to optimally, or near-optimally, combine what is known about the ships motion - the governing dynamics- with new information that is being made available - the measurements. This more accurate picture can then be used to control the system. The relationship between estimation and control theory, as we will see, goes beyond their connection in application and is fundamental in the underpinnings of these areas.

In this thesis, we introduce the theoretical basis for minimum energy estimation. In Chapter 2 we give an overview of optimal control methods, focusing on the Pontryagin minimum

principle and Bellman's dynamic programming. In Chapter 3, we use the latter method in the context of optimal estimation. Although this requires a leap in logic as the estimation errors become the "controls", the optimal control techniques work well in this area. The formulations of the minimum energy estimation are given in both continuous and discrete time-settings; the consistency between both formulations is also addressed. The derivation of the continuous time estimator dynamics by Mortensen in [75] and then extended to viscosity solutions by Krener in [58] is detailed in Section 3.3. The Hamilton-Jacobi-Bellman equation emerges as crucial to solving optimal estimation problems. Due to the difficulties in solving it, the myriad of numerical methods that exist for doing so are discussed in Chapter 4. Finally in Chapter 5, simulations of the Kalman filter applied to linear systems are used to show how early or late discretization of estimation dynamics can influence estimation accuracy.

Chapter 2

Overview of Optimal Control

This overview of optimal control theory is based on [67] and [37]. The latter text and accompanying course were my first introduction to optimal control.

2.1 The Basic Problem

Systems of ordinary differential equations (ODEs) are used to capture the dynamics, or the evolution, of a specific system that we are studying. These could include machines and processes in engineering or population and social dynamics in biology, to only mention a few. In general ODEs are written as:

$$\begin{cases} \dot{x}(t) = f(x(t)) & (t > 0) \\ x(0) = x_0. \end{cases}$$

Here $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a function and $x(\cdot)$ is the state of the system as it evolves and has the initial point $x_0 \in \mathbb{R}^n$.

In some cases, it is useful to include a control in the dynamics. For example, to model a pandemic we may want to introduce a control term representing vaccines or we may introduce fishing as a control in a fish population model. We could also be interested in the steering in a spacecraft navigation problem.

Most often, we want to have the control vary over time rather than be a fixed value, we include this in the model like so:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)) & (t > 0) \\ x(0) = x_0 \end{cases} \tag{2.1}$$

where $u : [0, \infty] \rightarrow \mathcal{A}$ is the control, a time dependent function with values in $\mathcal{A} \subset \mathbb{R}^m$, the admissible set. In other situations, $u(\cdot)$ can also represent disturbances or other external signals to the system.

We now mention the conditions that imply this problem is well-posed. We state assumptions that could be relaxed but in practice are appropriate. We assume f to be continuous in t and u and C^1 in x . We assume f_x to be continuous in t and u , here f_x is the Jacobian matrix of partial derivatives of f with respect to x . Finally, we assume $u(\cdot)$ to be piecewise continuous as a function of t .

See section 3.3.1 in [67] for a more detailed discussion of nuances that exist in the assumptions.

2.2 The Cost Functional

Not only are we interested in controlling a system, but want to do so in an optimal way. To work in optimal control we need a criterium. We introduce a payoff or cost functional:

$$J(t_0, x(\cdot), u(\cdot)) := \int_{t_0}^T L(x(t), u(t))dt + g(x(T))$$

subject to the control dynamics (2.1). For simplicity, we will interpret $J(\cdot)$ as a cost functional to be minimized but a change in signs turns it into a payoff or reward that should be maximized, it is simply a matter of goals and perspectives. This functional associates a certain cost with each possible path which itself is determined by the set of admissible control values. Note $L : \mathbb{R}^n \times \mathcal{A} \rightarrow \mathbb{R}$ is the running cost and $g : \mathbb{R}^n \rightarrow \mathbb{R}$ is the terminal cost and that T is the terminal time, this can either be fixed or free and $x(T)$ is the terminal state which can also be fixed or free.

Let's illustrate with an example. Imagine we are modeling the path of a rocket in outer space, we want it to follow a specific trajectory, $x(t)$, and to do so we can use some thrust, $u(t)$. The cost functional will penalize straying too far from the desired trajectory as well as having too much fuel consumption. The problem may include a terminal state, say a destination for the rocket or it may just specify terminal time, or both. The problem will then be solved by the set of controls which minimize the cost functional and adhere to the terminal constraints if there are any.

It is important to understand that the cost functional is chosen to fit the specific goals for a given problem or system. There are minimum time problems where the goal is to move the system from an initial state to a final state in the least amount of time, the design of

that cost is quite straightforward since we just want to minimize the elapsed time. Other problems may be more ambiguously posed and have many possible cost functions. For example the problem statement: “Maintain the position and velocity of the system near zero with a small expenditure of control energy” doesn’t yield a unique cost function. The designer may then have to try a variety of cost functions and choose one [55].

2.3 A Brief Introduction to Calculus of Variations

We have set the stage by introducing the basic problems in optimal control, we would also like to introduce the principal tools used to solve these problems. We thus take a brief detour in our introduction of optimal control theory to briefly introduce calculus of variations. Calculus of variations will serve as a stepping stone to the general ideas behind the Pontryagin Minimum Principle (PMP). Pontryagin’s Minimum Principle is a central idea in solving optimal control problems, so using calculus of variations as a motivation for it will be worth it. Readers may have been introduced to the PMP as the Pontryagin maximum principle; this is the same theory except for a change of signs when interpreting the cost function as a payoff. Since we are interested in minimization problems, we will just introduce the minimum principle.

Given a function $L : [t_0, T] \times \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ - often referred to as a Lagrangian to honor Lagrange’s contribution - which depends on time t , position x and velocity \dot{x} and where T is terminal time, the basic problem in calculus of variations is to find the optimal curve $x^* \in \mathcal{X}$ that minimizes the functional

$$J(t_0, x(\cdot), \dot{x}(\cdot)) := \int_{t_0}^T L(t, x(t), \dot{x}(t))dt + g(x(T))$$

with the admissible class $\mathcal{X} = \{x : [t_0, T] \rightarrow \mathbb{R} \mid x(\cdot)$ is continuous and piecewise continuously differentiable $\}$ and also possibly satisfies some initial condition $x(t_0) = x_0$, or terminal condition and $x(T) = x_1$. Note that even though x and \dot{x} are the position and velocity, the Lagrangian L is considered a function of three independent variables, often it will be denoted $L(x, y, z)$ or $L(t, x, v)$ but in our work, $v = \dot{x}$.

Definition 1. The gradient of V is denoted:

$$\nabla_x V(t, x) = \begin{bmatrix} \frac{\partial V}{\partial x_1}(t, x) \\ \vdots \\ \frac{\partial V}{\partial x_n}(t, x) \end{bmatrix}.$$

┘

Theorem 1 (Euler-Lagrange Equations [37]). *If $x^*(\cdot)$ solves the calculus of variations problem and $L \in C^1$, then $x^*(\cdot)$ is a solution to the Euler-Lagrange (E-L) differential equation:*

$$\frac{d}{dt}[\nabla_v L(t, x^*(t), \dot{x}^*(t))] = \nabla_x L(t, x^*(t), \dot{x}^*(t)),$$

where $L = L(t, x, v)$ and $\nabla_x L = (\frac{dL}{dx_1}, \frac{dL}{dx_2}, \dots, \frac{dL}{dx_n})$ and $\nabla_v L = (\frac{dL}{dv_1}, \frac{dL}{dv_2}, \dots, \frac{dL}{dv_n})$.

Solutions x^* are called stationary points of L .

This theorem is relevant because it shows that, if they exist, solutions to the calculus of variations problem can be found by first solving the E-L equations. The idea of being able to find solutions to one problem by solving a related one is key in mathematics.

We will continue the introduction to calculus of variations by defining $p(\cdot)$, the generalized momentum, as:

$$p(t) := \nabla_v L(t, x(t), \dot{x}(t)) \quad t \in [0, T].$$

With p we will be able to re-write the E-L equations as a systems first order ODEs. Define the *Hamiltonian* of the dynamical system:

$$H(x, p) := p \cdot f(x, p) + L(t, x, f(x, p)).$$

Theorem 2 (Hamilton's Equations [37]). *Let $x^*(\cdot)$ solve the E-L equations and define $p(\cdot)$ as above, then the pair $(x(\cdot), p(\cdot))$ solves Hamilton's equations:*

$$\begin{cases} \dot{x}(t) = \nabla_p H(x(t), p(t)) \\ \dot{p}(t) = -\nabla_x H(x(t), p(t)). \end{cases}$$

2.4 Pontryagin Minimum Principle

We now introduce a central result in optimal control theory, the Pontryagin minimum principle.

In the 1950's, the Soviet mathematician Lev Pontryagin started working on minimum-time interception problems. Nonlinear due to the aircraft dynamics, these problems aim to “find the optimal trajectory of an aircraft that is to be steered from a given cruise position into a favorable position against an attacking hostile aircraft.” His work in the area led to these central results [84].

Fixed Time, Free Endpoint

We first introduce PMP for a fixed time, free endpoint problem. We want to find a control $u^*(\cdot) \in \mathcal{A}$, the admissible set, where $u(\cdot) : [0, \infty) \rightarrow \mathbf{A}$ is a function which for the system of ODEs:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)) & t \in [t_0, T] \\ x(t_0) = x_0 \end{cases} \quad (2.2)$$

minimizes the cost:

$$J(t_0, x(\cdot), u(\cdot)) = \int_{t_0}^T L(x(t), u(t)) dt + g(x(T)). \quad (2.3)$$

We are minimizing some cost with a set terminal time but no target end state.

Before stating PMP, which says that there exists a function $p(\cdot)$ which together with $x(\cdot)$ solves Hamilton's ODEs as above, we must define the control theory Hamiltonian; it is very similar to the Hamiltonian defined in the context of calculus of variations. It is the function $H(x, p, u) := f(x, u) \cdot p + L(t, x, u)$. Note we have already formulated it for minimization problems where we are minimizing the cost function in the form (2.3).

For this type of problem, PMP is stated as follows:

Theorem 3 (Pontryagin minimum principle for fixed time, free endpoint problems). *Let $u^*(\cdot)$ be optimal for the ODE and the cost function and let $x^*(\cdot)$ be the corresponding state trajectory, then there exists $p^*(\cdot) : [0, T] \rightarrow \mathbb{R}^n$ such that*

$$\dot{x}^*(t) = \nabla_p H(x^*(t), p^*(t), u(t)) \quad (\text{ODE})$$

$$\dot{p}^*(t) = -\nabla_x H(x^*(t), p^*(t), u(t)) \quad (\text{ADJ})$$

and

$$H(x^*(t), p^*(t), u^*(t)) = \min_{u \in \mathcal{A}} H(x^*(t), p^*(t), u) \quad (0 \leq t \leq T) \quad (\text{MIN})$$

with $H(x^*(t), p^*(t), u^*(t))$ constant, or time invariant, and with the terminal condition:

$$p^*(T) = \nabla g(x^*(T)). \quad (\text{T})$$

It is hopefully more clear as to why calculus of variations was reviewed since (ODE) and (ADJ), often called the canonical equations, have a clear resemblance to Hamilton's equations in section 2.3.

Free Time, Fixed Endpoint

Now, we give the formulation of PMP for free time, fixed endpoint problems which has some slight differences. The main one being that we introduce a target end point x_1 .

The cost functional changes to:

$$J(\tau, x(\cdot), u(\cdot)) = \int_{t_0}^{\tau} L(x(t), u(t))dt + g(x(\tau)), \quad (2.4)$$

since the terminal time is not known. Instead τ takes its place, where τ is the time when the state $x(\cdot)$ first reaches x_1 . We want to find a control $u^*(\cdot) \in \mathcal{A}$ is a function which for the system of ODEs:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)) & (t > 0) \\ x(t_0) = x_0 \\ x(T) = x_1 \end{cases} \quad (2.5)$$

minimizes the cost (2.4). Meaning we are minimizing some cost with a set terminal time but no target end state, as the system of ODEs doesn't have a transversality condition.

Theorem 4 (Pontryagin minimum principle for free time, fixed endpoint problems). *Let $u^*(\cdot)$ be optimal for the ODE and the cost function and let $x^*(\cdot)$ be the corresponding state trajectory, then there exists $p^*(\cdot) : [0, \tau^*] \rightarrow \mathbb{R}^n$ such that*

$$\dot{x}^*(t) = \nabla_p H(x^*(t), p^*(t), u^*(t)), \quad x(\tau^*) = x_1 \quad (\text{ODE})$$

$$\dot{p}^*(t) = -\nabla_x H(x^*(t), p^*(t), u^*(t)) \quad (\text{ADJ})$$

and

$$H(x^*(t), p^*(t), u^*(t)) = \min_{u \in \mathcal{A}} H(x^*(t), p^*(t), u) \quad (0 \leq t \leq \tau) \quad (\text{MIN})$$

where τ denotes the first time the target state is obtained by $x^*(\cdot)$. $p^*(\cdot)$ is called the costate.

Remark 1. In some presentations, the abnormal multiplier q is explicitly written in $H(x, p, u) := f(x, u) \cdot p - L(t, x, u) \cdot q$. In the degenerate case, $q = 0$; however, we are assuming $q = 1$ in our work. ┘

Other formulations or variations of PMP exist for problems with transversality conditions or state constraints, details can be found in detail in [37].

Note that PMP yields an open-loop control law $u(\cdot)$ meaning it does not depend on the current state $x(\cdot)$. An open-loop control law is computed once for a specific initial condition and not adjusted “on the go” based on real-time feedback from the state.

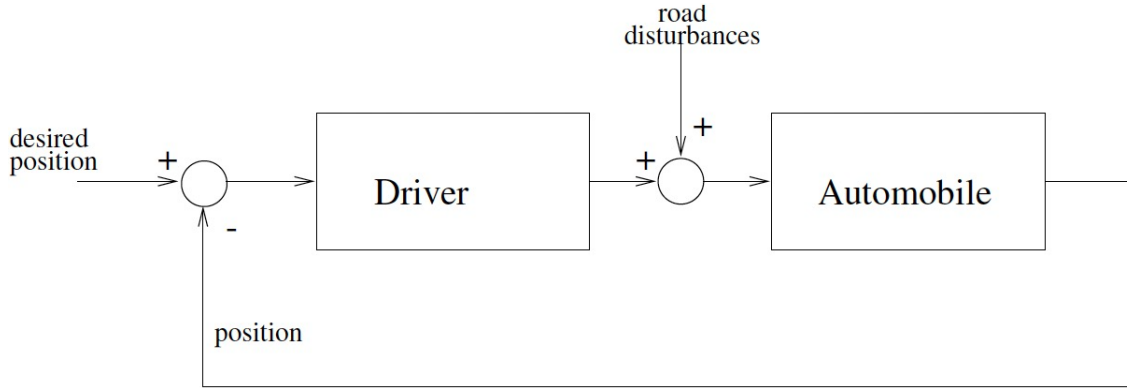


Figure 2.1: Driver Automobile system, replicated from figure 1.1 in [74].

To illustrate this let's present an example from [74]. In the system represented by Figure 2.1, the driver controls the automobile and steers it towards its desired position. In an open-loop system, the driver would choose a direction to steer towards and then keep that direction without taking into account the changing position of the car, the road, or any other inputs. In a feedback system, or closed-loop system, the driver is continuously taking in the changes in the road, the updated position in the car in order to fine-tune the steering to reach the desired position at each time. Open-loop systems are imprecise, unable to adapt and have limited real-world applications. Closed-loop systems, on the other hand, are everywhere from one's daily commute to blood-sugar regulation via insulin injection to the self-adjusting nature of pricing dynamics.

2.5 Finite-Time Linear-Quadratic Regulator Problem Solved Using Pontryagin's Minimum Principle

A common type of problem where the control dynamics are linear and the cost functional is quadratic is referred to as an LQR problem. We adapt the presentation from [74].

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ x(0) = x_0 \end{cases} \quad (2.6)$$

where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$.

Definition 2. A symmetric matrix $G \in \mathbb{R}^{n \times n}$ is positive semi-definite or non-negative, if

$$a^\top G a \geq 0$$

for all $a \in \mathbb{R}^n$. ┘

We also have the quadratic cost:

$$J(T; u) = \frac{1}{2} \int_0^T x(t)^\top Q x(t) + u(t)^\top R u(t) dt + g(x(T))$$

with Q and R being symmetric, positive semi-definite matrices.

The optimal control problem can be written concisely as:

$$\inf_{u \in (0, T; \mathbb{R}^m)} J(T; u)$$

subject to the ODE (2.6). The minimizing control u^* is the optimal control.

The control Hamiltonian is

$$\begin{aligned} H(x(t), p(t), u(t)) &= f(x(t), u(t)) \cdot p(t) + L(x(t), u(t)) \\ &= (Ax(t) + Bu(t))p(t) + \frac{1}{2}x(t)^\top Q x(t) + \frac{1}{2}u(t)^\top R u(t). \end{aligned}$$

Before computing the optimality equations for this problem let's introduce the following lemma.

Lemma 1. For a matrix $A \in \mathbb{R}^{n \times n}$ and vector $x \in \mathbb{R}^n$,

$$\frac{d(x^\top Ax)}{dx} = x^\top (A + A^\top).$$

In the case that A is symmetric, as we have in many of our problems, we obtain:

$$\frac{d(x^\top Ax)}{dx} = 2x^\top A.$$

The proof, as presented in [74] is as follows:

Proof. Let $F(x)$ be a scalar valued function such that $F(x) = x^\top Ax$. For arbitrary vectors $x, h \in \mathbb{R}^n$,

$$\begin{aligned} F(x+h) - F(x) &= x^\top Ah + h^\top Ax + h^\top Ah \\ &= x^\top Ah + x^\top A^\top h + h^\top Ah \\ &= h^\top (A + A^\top)x + h^\top Ah. \end{aligned}$$

Since

$$|h^\top Ah| \leq \|x\| \|A\| \|x\|,$$

then

$$\lim_{\|h\| \rightarrow 0} \frac{|h^\top Ah|}{\|h\|} = 0.$$

And thus:

$$\lim_{\|h\| \rightarrow 0} \frac{|h^\top (A + A^\top)x + h^\top Ah|}{\|h\|} = x^\top (A + A^\top).$$

So $F(x)$ is differentiable with derivative $x^\top (A + A^\top)$.

And if A is symmetric then $(A + A^\top) = 2A$. □

Now, it is easy to write the partials of f and r which will be used in the optimality equations:

$$f_x = A, f_u = B, L_x = x^\top Q, L_u = u^\top R.$$

So we have (ADJ):

$$\begin{aligned} \dot{p}(t) &= -\nabla_x H(x(t), p(t), u(t)) \\ &= Qx(t) - A^\top p(t). \end{aligned}$$

And recall we also have (ODE):

$$\begin{aligned} \dot{x}(t) &= \nabla_p H(x(t), p(t), u(t)) \\ &= Ax(t) + Bu(t). \end{aligned}$$

And if u^* is the minimizing control, (MIN) yields:

$$\begin{aligned} H_{u^*} &= 0 \\ (u^*(t))^\top R - p(t)^\top B &= 0. \end{aligned}$$

Solving to find the optimal control u^* which minimizes $H(x, p, u)$ yields:

$$\begin{aligned} (u^*(t))^\top R - p(t)^\top B &= 0 \\ Ru^*(t) &= p(t)^\top B \\ u^*(t) &= R^{-1} B^\top p(t). \end{aligned}$$

or if we set $z(t) = -p(t)$, we obtain:

$$u^*(t) = -R^{-1} B^\top z(t).$$

Applying $z(t) = -p(t)$ to

$$\dot{p}(t) = Qx(t) - A^\top p(t) \quad (2.7)$$

yields:

$$\dot{z}(t) = -Qx(t) - A^\top z(t).$$

And applying $u^*(t) = R^{-1}B^\top p(t)$ to

$$\dot{x}(t) = Ax(t) + Bu(t) \quad (2.8)$$

yields:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu^*(t) \\ &= Ax(t) + B(-R^{-1}B^\top z(t)) \\ &= Ax(t) - BR^{-1}B^\top z(t), \end{aligned}$$

with the initial and terminal conditions:

$$x(0) = x_0, \quad z(T) = 0.$$

If $x^*(t)$ is the trajectory of the system with optimal control $u^*(t)$ and if $\exp(At)$ is the matrix exponential of At , we can write:

$$x^*(t) = \exp(At)x_0 + \int_0^t \exp A(t-s)Bu(s)ds. \quad (2.9)$$

and

$$z(t) = \int_t^T \exp A^\top(s-t)Qx^*(s)ds \quad (2.10)$$

giving the optimal control:

$$u^*(t) = -R^{-1}B^\top \left[\int_t^T \exp A^\top(s-t)Qx^*(s)ds \right]. \quad (2.11)$$

This expression, however, involves future terms of x^* which itself depends on u^* . So we will now rewrite (2.11) to only involve the current state and not future ones.

Let's define:

$$P(t;T)x^*(t) := \int_t^T \exp A^\top(s-t)Qx^*(s)ds. \quad (2.12)$$

From (2.10) and (2.11) we can write:

$$\begin{aligned} z^*(t) &= P(t; T)x^*(t) \\ u^*(t) &= -R^{-1}B^\top P(t; T)x^*(t). \end{aligned} \quad (2.13)$$

We differentiate (2.12) with respect to t :

$$\dot{P}(t; T)x^*(t) + P(t; T)\dot{x}^*(t) = -Qx^*(t) - A^\top \int_t^T \exp A^\top(s-t)Qx^*(s)ds. \quad (2.14)$$

Differentiating (2.9) and using (2.11) we have:

$$\dot{x}^*(t) = Ax^*(t) - BR^{-1}B^\top P(t; T)x^*(t). \quad (2.15)$$

Substituting (2.15) into (2.14) and using (2.12) on the right hand side, we obtain:

$$\begin{aligned} \dot{P}(t; T)x^*(t) + P(t; T)Ax^*(t) - P(t; T)BR^{-1}B^\top P(t; T)x^*(t) = \\ -Qx^*(t) - A^\top P(t; T)x^*(t). \end{aligned} \quad (2.16)$$

Factoring by $x^*(t)$, then dividing by it, and moving all terms to one side, we will finally obtaining:

$$\begin{aligned} \dot{P}(t; T) + P(t; T)A + A^\top P(t; T) - P(t; T)BR^{-1}B^\top P(t; T) + Q = 0 \\ P(T; T) = 0. \end{aligned}$$

This is a differential Riccati equation which can be solved to obtain the optimal control formulation:

$$u^*(t) = R^{-1}B^\top P(t; T)x^*(t).$$

For more details on this proof, see Theorem 10.3 and the accompanying proof in [74].

2.6 Dynamic Programming and the Hamilton-Jacobi-Bellman Equation

We will now step away from PMP to introduce dynamic programming and the Hamilton-Jacobi-Bellman equation. This is a different approach to solving optimal control problems which was developed independently from the PMP. As stated, PMP was developed in the Soviet Union. On the other side of the Iron Curtain, another method to solve these kinds of problems was being developed. In the 1950's, Richard Bellman worked at the RAND

corporation, a research and development think tank in Santa Monica, California created by the U.S. Air Force to collaborate closely with the U.S. government and military. Like Pontryagin, he was researching minimum time interception problems [84].

In Bellman’s 1957 book “Dynamic Programming”, he writes “In place of determining the optimal sequence of decisions from the fixed state of the system, we wish to determine the optimal decision to be made at any state of the system. Only if we know the latter, do we understand the intrinsic structure of the solution.” In section 2.3, on calculus of variations, we saw that often, in mathematics, we are able to solve a problem by solving a similar one, or generalizing it. In dynamic programming, we encounter a different approach that is central to problem solving which is to break down a large problem into smaller chunks. Let’s again look at the problem of finding an optimal control for the system:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)) & (t > 0) \\ x(t_0) = x_0 \end{cases} \quad (2.17)$$

with the cost functional:

$$J(t_0, x(\cdot), u(\cdot)) := \int_{t_0}^T L(x(t), u(t))dt + g(x(T)).$$

Instead of solving the control ODE problem with the cost functional with a fixed starting time, we consider the following cost functional with varying start time t

$$J(t, x(\cdot), u(\cdot)) := \int_t^T L(x(t), u(t))dt + g(x(T)), \quad t \in [t_0, T].$$

We define the value function:

$$V(t, x) := \inf_{u \in (t, T; \mathbb{R}^m)} J(t, x, u) \quad (2.18)$$

as the lowest cost possible starting at time t with state x . Notice we use inf instead of min as we don’t know if this optimal cost, which is given by an optimal control, actually exists. The central idea in dynamic programming is to solve the collection of minimization problems for many different times t and establish a relationship between them in order to solve the more general problem with the start time cost functional.

Theorem 5 (Bellman’s Optimality Principle [67]). *For every pair $(t, x) \in [t_0, t_1] \times \mathbb{R}^n$ and every $\Delta t \in (0, t_1 - t]$, the value function V , as defined in (2.18), satisfies the relation*

$$V(t, x) = \inf_{u_{[t, t+\Delta t]}} \left\{ \int_t^{t+\Delta t} L(s, x(s), u(s))ds + V(t + \Delta t, x(t + \Delta t)) \right\}$$

where $x(\cdot)$ on the right-hand side is the state trajectory corresponding to the control $u_{[t,t+\Delta t]}$ and satisfying $x(t) = x$.

We will first present the Hamilton-Jacobi-Bellman (HJB) equation and then a heuristic proof. See section 3.11 in [55] and Theorem 5.1.1 in Chapter 5 of [37] for formal proofs of the HJB, and Theorem 2 section 10.3 in [36] or section 5.13 in [67] for different presentations of heuristic proofs of the HJB.

Theorem 6 (Hamilton-Jacobi-Bellman Equation). *If the value function $V \in C^1([0, T], \mathbb{R}^n)$, it solves the following partial differential equation:*

$$-\frac{\partial V}{\partial t}(t, x) = \inf_{u \in \mathcal{A}} \left\{ L(t, x, u) + f(x, u) \cdot \nabla_x V(t, x) \right\} \quad (2.19)$$

with the terminal condition $V(T, x) = g(x)$.

Let's present a heuristic proof of the HJB equation.

Proof. By definition of the value function:

$$V(t, x) = \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_t^T L(\tau, x(\tau), u(\tau)) d\tau + g(x(T)) \right\},$$

where $x(\tau)$ is evolving according to the dynamics (2.17) and u is in the set of admissible controls $\mathcal{A} = \{u(\cdot) : [0, T] \rightarrow A \mid u(\cdot) \text{ is piecewise continuous}\}$. The set A will often be the set of real numbers but could be a subset of it for a given problem.

Now, using Bellman's principle of optimality, we can split up the bounds of the integral in the value function as:

$$\begin{aligned} V(t, x) &= \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau \right\} \\ &\quad + \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_{t+\Delta t}^T L(\tau, x(\tau), u(\tau)) d\tau + g(x(T)) \right\}. \end{aligned}$$

Notice the term:

$$\inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_{t+\Delta t}^T L(\tau, x(\tau), u(\tau)) d\tau + g(x(T)) \right\} = V(t + \Delta t, x(t + \Delta t)).$$

So we re-write the value function as:

$$V(t, x) = \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau \right\} + V(t + \Delta t, x(t + \Delta t)).$$

Move the $V(t, x)$ to the right hand side and re-arrange:

$$0 = \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau \right\} + V(t + \Delta t, x(t + \Delta t)) - V(t, x).$$

Divide by Δt :

$$0 = \frac{V(t + \Delta t, x(t + \Delta t)) - V(t, x)}{\Delta t} + \frac{1}{\Delta t} \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau \right\}.$$

Now let's take $\Delta t \rightarrow 0$.

$$\begin{aligned} 0 &= \lim_{\Delta t \rightarrow 0} \left\{ \frac{V(t + \Delta t, x(t + \Delta t)) - V(t, x)}{\Delta t} + \frac{1}{\Delta t} \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau \right\} \right\}, \\ 0 &= \lim_{\Delta t \rightarrow 0} \frac{V(t + \Delta t, x(t + \Delta t)) - V(t, x)}{\Delta t} + \lim_{\Delta t \rightarrow 0} \frac{1}{\Delta t} \inf_{u(\cdot) \in \mathcal{A}} \left\{ \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau \right\}. \end{aligned}$$

We can move $\lim_{\Delta t \rightarrow 0} \frac{1}{\Delta t}$ inside the inf because of continuity of $L(\cdot)$ and compactness of \mathcal{A} :

$$0 = \lim_{\Delta t \rightarrow 0} \frac{V(t + \Delta t, x(t + \Delta t)) - V(t, x)}{\Delta t} + \inf_{u(\cdot) \in \mathcal{A}} \left\{ \lim_{\Delta t \rightarrow 0} \frac{1}{\Delta t} \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau \right\}.$$

Notice the term:

$$\lim_{\Delta t \rightarrow 0} \frac{V(t + \Delta t, x(t + \Delta t)) - V(t, x)}{\Delta t} = \frac{\partial V}{\partial t}(t, x) + \nabla_x V(t, x) \cdot \dot{x}.$$

This is the total derivative of the value function V , we used the chain rule for V with respect to both t and x .

Meanwhile the expression:

$$\lim_{\Delta t \rightarrow 0} \frac{1}{\Delta t} \int_t^{t+\Delta t} L(\tau, x(\tau), u(\tau)) d\tau = L(t, x(t), u(t)).$$

Thus we obtain:

$$0 = \frac{\partial V}{\partial t}(x, t) + \nabla_x V(t, x) \cdot \dot{x} + \inf_{u(\cdot) \in \mathcal{A}} \left\{ L(\tau, x(\tau), u(\tau)) \right\},$$

We can include the $\nabla_x V(t, x) \cdot \dot{x}$ in the inf as well since \dot{x} depends on the control u .

We obtain the HJB:

$$0 = \frac{\partial V}{\partial t}(x, t) + \inf_{u(\cdot) \in \mathcal{A}} \left\{ \nabla_x V(t, x) \cdot \dot{x} + L(t, x(t), u(t)) \right\}$$

thus we have derived the HJB. □

Remark 2. Often, the HJB will be presented having replaced \dot{x} by $f(x, u)$:

$$0 = \frac{\partial V}{\partial t}(x, t) + \inf_{u(\cdot) \in \mathcal{A}} \left\{ \nabla_x V(t, x) \cdot f(x, u) + L(t, x(t), u(t)) \right\},$$

this can be done when the dynamics are in the form of (2.17). ┘

The Hamilton-Jacobi-Bellman equation is a first-order PDE that determines the value function, $V(t, x)$. This function takes an initial state (t, x) as input and outputs the optimal value of the cost function.

The HJB and the control Hamiltonian are connected in that the HJB can be expressed in terms of the optimal Hamiltonian H_* .

It is important to note that the theory of the HJB equation often assumes the value function V is sufficiently smooth. However, in many practical cases, V may lack the required regularity. To address this, the concept of viscosity solutions has been introduced. Viscosity solutions allow us to make sense of and solve the HJB equation even when V is not smooth. Since this thesis will reference viscosity solutions in later sections, we provide a brief introduction to them here.

2.6.1 Viscosity Solutions to Partial Differential Equations

Before we can introduce viscosity solutions, we must first introduce some concepts from non-smooth analysis. Let $v : \mathbb{R}^n \rightarrow \mathbb{R}$ be a continuous function.

Definition 3. A vector ξ is called a super-differential of v at a point x if for all δ near x :

$$v(\delta) \leq v(x) + \langle \xi, \delta - x \rangle + o(|\delta - x|)$$

where $o(\alpha)$ represents higher-order, i.e. $\lim_{\alpha \rightarrow 0} \frac{o(\alpha)}{\alpha} = 0$ and $\langle x, y \rangle$ is the dot product of vectors x, y . ┘

The geometric interpretation of the above definition is that ξ is a super-differential of v if the graph of the function $\varphi : \delta \rightarrow v(x) + \langle \xi, \delta - x \rangle$, which has gradient $\nabla \varphi(x) = \xi$ and has the value $\varphi(x) = v(x)$ at x , lies above the graph of v , or is tangent to it, near x . The function φ is called a test function. A super-differential is not usually not unique, and thus we denote $D^+v(x)$ the set of super-differentials of v at x .

If, instead, we require the graph of this function to be below, or tangent to, that of $v(x)$, we use the following nomenclature.

Definition 4. A vector ξ is called a sub-differential of v at a point x if for all δ near x the following relations holds:

$$v(\delta) \geq v(x) + \langle \xi, \delta - x \rangle - o(|\delta - x|).$$

┘

And we denote $D^-v(x)$ the set of sub-differentials of v at x .
If v is differentiable at the point x :

$$D^+v(x) = D^-v(x) = \nabla v(x).$$

For a proof, see section 5.3.1 in [67].

Now, let's consider a PDE in the general form:

$$F(x, v(x), \nabla v(x)) = 0 \tag{2.20}$$

where $F : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}$ is a continuous function.

Definition 5. A viscosity subsolution of the PDE (2.20) is a continuous function $v : \mathbb{R}^n \rightarrow \mathbb{R}$ such that

$$F(x, v, \xi) \leq 0 \quad \forall \xi \in D^+v(x), \forall x.$$

┘

In other words, for a C^1 function φ such that $\varphi - v$ has a local minimum at x , we must have, at every x :

$$F(x, v(x), \nabla \varphi(x)) \leq 0.$$

Definition 6. A viscosity supersolution of the PDE (2.20) is a continuous function $v : \mathbb{R}^n \rightarrow \mathbb{R}$ such that

$$F(x, v, \xi) \geq 0 \quad \forall \xi \in D^-v(x), \forall x.$$

┘

In other words, for a C^1 function φ such that $\varphi - v$ has a local maximum at x , we must have, at every x :

$$F(x, v(x), \nabla \varphi(x)) \geq 0.$$

We now use both definitions to define a viscosity solution.

Definition 7. The function v is a viscosity solution if it is both a viscosity subsolution and viscosity supersolution.

┘

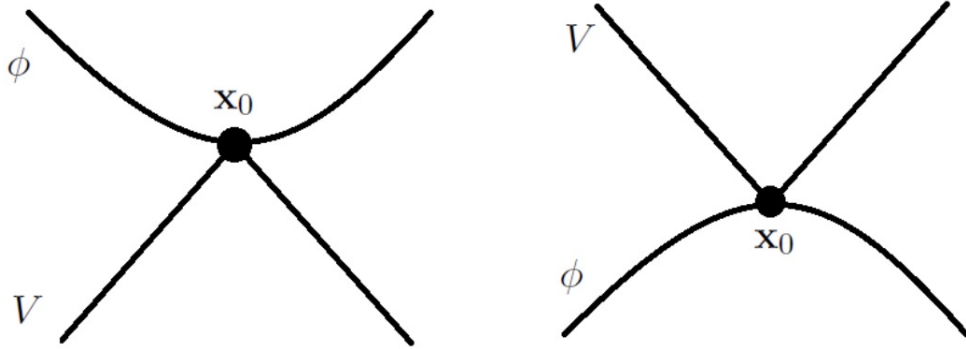


Figure 2.2: Subsolution and supersolution at a point x_0 where ∇V is not defined. Figure 3.4 replicated from [94].

The intuition behind this weaker type of solutions for PDEs is the following. Suppose that we have a candidate solution function v that is not differentiable throughout the domain. We can use a smooth test function φ , it “touches” v from below and above at all points including where v is non differentiable. At points where v is differentiable, the PDE holds in the classical sense, but at points where it is non-differentiable, $\nabla\varphi$ i.e. ξ is used instead. In short, it allows non-smooth functions to weakly solve PDEs, where they are differentiable they are the solutions and where they are not the gradient of an appropriate test function is used instead.

There are a variety of equivalent definitions of viscosity solutions, a discussion can be found in Section II.4 in [40].

The first definition of this type of solutions was given in 1980 by Evans in [35], but not given the name viscosity solution until 1983 by Crandall and Lions in [27]. The definitions and properties were then further refined by all three authors in [26]. For more details on viscosity solutions see [100].

2.7 Linear-Quadratic Regulator Problem Solved Using Dynamic Programming

Using the same dynamics and cost function as in section 2.5 we will show how the LQR problem can be solved using Bellman’s approach.

Recall the HJB is:

$$-\frac{\partial V}{\partial t}(x, t) = \inf_{u \in \mathcal{A}} \left\{ \frac{\partial V}{\partial x}(x, t) \cdot f(x, u) + L(t, x, u) \right\}$$

with the terminal condition:

$$V(T, x) = g(x).$$

In this example:

$$\begin{aligned} f(x, u) &= Ax + Bu \\ L(t, x, u) &= x^\top Qx + u^\top Ru \\ g(x) &= x^\top Wx. \end{aligned}$$

So we obtain the following HJB:

$$0 = \frac{\partial V}{\partial t} + \inf_{u \in \mathcal{A}} \left\{ \frac{\partial V}{\partial x}(Ax + Bu) + x^\top Qx + u^\top Ru \right\} \quad (2.21)$$

with the terminal condition:

$$V(T, x) = x^\top Wx. \quad (2.22)$$

We use the quadratic form of terminal condition (2.22) to guess that the value function will have a quadratic form as well:

$$V(t, x) = x^\top P(t)x$$

where P is symmetric and $P(T) = W$.

Now we compute the partial derivatives of the value function:

$$\frac{\partial V}{\partial t} = x^\top \dot{P}(t)x, \quad \frac{\partial V}{\partial x} = 2x^\top P(t).$$

Substituting this into (2.21):

$$0 = x^\top \dot{P}x + \inf_{u \in \mathcal{A}} \left\{ 2x^\top P(Ax + Bu) + x^\top Qx + u^\top Ru \right\}.$$

To minimize inside the brackets we take the derivative w.r.t u :

$$\begin{aligned} \frac{\partial}{\partial u}(2x^\top PBu + u^\top Ru) &= 0 \\ 2B^\top Px + 2Ru &= 0. \end{aligned}$$

We finally solve for u :

$$\begin{aligned} Ru &= -B^\top Px \\ u &= -R^{-1}B^\top Px. \end{aligned}$$

We have found the optimal control law $u = -R^{-1}B^\top Px$, let's substitute it into (2.21):

$$\begin{aligned} 0 &= x^\top \dot{P}x + 2x^\top P(Ax + B(-R^{-1}B^\top Px)) + x^\top Qx \\ &\quad + (-R^{-1}B^\top Px)^\top R(-R^{-1}B^\top Px). \end{aligned}$$

We simplify:

$$\begin{aligned} 0 &= x^\top \dot{P}x + x^\top PAx + x^\top A^\top Px + 2x^\top PB(-R^{-1}B^\top Px) + x^\top Qx \\ &\quad + x^\top PB(R^{-1})^\top B^\top Px. \end{aligned}$$

We gather like terms:

$$0 = x^\top \left(\dot{P} + PA + A^\top P - PBR^{-1}B^\top P + Q \right) x.$$

And as the last step, we show how this leads to a differential Riccati equation:

$$\begin{aligned} 0 &= x^\top \left(\dot{P} + PA + A^\top P - PBR^{-1}B^\top P + Q \right) x \\ 0 &= \dot{P} + PA + A^\top P - PBR^{-1}B^\top P + Q \\ \dot{P} &= -PA - A^\top P + PBR^{-1}B^\top P - Q, \end{aligned} \tag{DRE}$$

with the terminal condition $P(T) = W$.

If the infinite-horizon case is solved instead, the value function is time-invariant and $\frac{\partial V}{\partial t}$ vanishes. The resulting equation for P is the algebraic Riccati equation:

$$0 = PA + A^\top P - PBR^{-1}B^\top P + Q. \tag{ARE}$$

2.8 Discussion of Pontryagin Minimum Principle and Hamilton-Jacobi-Bellman Equation

We would now like to briefly discuss the differences and similarities between the optimality equations in PMP and the HJB in dynamic programming as they give results for the same type of problems.

The Pontryagin Minimum Principle provides a necessary condition for optimality. This means that any optimal trajectory must satisfy the PMP, though the converse is not necessarily true; non-optimal trajectories may also fulfill it. Specifically, PMP requires that, at each time t , the optimal control $u^*(t)$ must minimize the Hamiltonian $H(x^*(t), p^*(t), u)$. This condition is referred to as an open-loop specification because the optimal control $u^*(t)$ depends not only on the state $x^*(t)$ but also on the co-state $p^*(t)$, which evolves according to the system dynamics and adjoint equations.

The Hamilton-Jacobi-Bellman equation provides a necessary and sufficient condition for optimality under appropriate smoothness assumptions. Any solution $V(x)$ of the HJB equation corresponds to the value function of an optimal control problem, and the control law derived from $V(x)$ is guaranteed to be optimal.

One of the key advantages of solving optimal control problems through the HJB equation is the ability to obtain feedback controls. Feedback controls depend only on the current state $x^*(t)$, as the optimal control $u^*(t)$ is determined directly from the gradient of the value function V at $x^*(t)$. This enables the control law to dynamically adapt to real-time states of the system, making it robust to disturbances and uncertainties. The adaptability of feedback controls enhances stability and performance, even in the presence of model inaccuracies or external perturbations [67].

Computationally, PMP leads to solving an ODE with split boundary conditions, this is much easier to solve than the nonlinear HJB. The HJB admits classical solutions only for a small set of sufficiently smooth value functions; in most cases this does not occur leading to the reliance on numerical approximations. The HJB is subject to the “curse of dimensionality”, a term coined by Bellman in [12] which refers to the fact that the computational cost for solving the HJB increases exponentially with the dimensionality of the problem. This and numerical methods developed to circumvent it will be discussed in Chapter 4.

Chapter 3

Optimal State Estimation

Up to this point, we have focused on dynamical systems with a control input in order to introduce the techniques of optimal control. In such systems, the goal is to determine the best control strategy that drives the system to a desired state while minimizing a cost function. However, control decisions rely on accurate knowledge of the system's current state, which is often not directly measurable. This brings us to the important problem of state estimation, where the challenge is to estimate the system's true state based on noisy or incomplete measurements.

In an ideal system, all states would be measured with full accuracy, however, in practical applications, it is rarely feasible to measure all state variables directly, particularly when dealing with complex or high-dimensional systems. Furthermore the measurements we do obtain are from sensors which are often imperfect or subject to random disturbances, and thus provide only partial views of the system's true state. As a result, estimation is essential for reconstructing the full state from partial, noisy, or indirect observations. The full estimated state can then be used to control the system.

For instance, in spacecraft navigation, sensors provide limited and noisy data, yet accurate state estimates are necessary for mission-critical control decisions. In this example, we have a model for the spacecraft. This model approximates the actual dynamics to some degree of accuracy, depending on factors like linearity assumptions and model order. Then we have measurements for the system; as discussed, the number of measurements and their accuracy may be limited. The goal of estimation is to use the model and the measurements we have in order to minimize the error between the estimated state and the actual state.

Estimation can be done in a stochastic or deterministic setting. If the state dynamics are defined by stochastic differential equations and the process and measurement noises are

assumed to be stochastic, the algorithm used for state estimation is usually called a filter [59]. The objective of state estimation in this case is to produce an estimate of the mean state and parameters using the data at hand. Often, the maximum likelihood estimator or a minimum variance estimator are used. It is important to note that both can be equivalent in certain cases [13]. In this scenario, the estimator is typically associated with a conditional expectation whereas covariance operators characterize the uncertainty.

If the model is a dynamical system and the "noises" are assumed to be deterministic - in which case we refer to them as disturbances - the resulting estimation algorithm is called an observer or estimator instead of filter. In this situation, the process and measurement disturbances are thought of as arbitrary "noises" rather than random ones. A norm characterizing the weight of the unknown, deterministic errors is introduced. The connections between the stochastic and deterministic paradigms have been shown in [9], [43].

Once the stochastic or deterministic framework has been decided upon, the estimation can be done in two ways. One is a variational strategy estimate, which computes the entire optimal trajectory in one go. This is usually done by solving an optimization problem often resulting in the need to solve a PDE. The other way is to use sequential estimators that take into account the discrepancy between the estimated trajectory and available measurements in order to update the estimate; a recursive procedure with a prediction step and a correction step is used.

The optimal observer for linear systems with Gaussian process and sensor noises is the well known Kalman filter [52]. This algorithm was used by NASA for the moon landing [71]. The term in the update equations which corrects the estimate according to new measurements, known as the Kalman gain, is computed by solving a Riccati equation which can be solved in a straightforward manner. Although the Kalman filter has been extended to nonlinear systems as the Extended Kalman Filter and the Unscented Kalman Filter [88], neither of these is optimal and may encounter stability issues depending on the system's nonlinearity and noise characteristics.

An optimal nonlinear estimate in the deterministic setting does exist: we will present the minimum energy estimate (MEE). Also known as the Mortensen observer or the maximum likelihood estimate, this observer was first introduced in [75]. As we will see, the gain for the MEE is obtained through solving the HJB equation which is notoriously difficult to do. The stability of error is also not guaranteed although stability results will be discussed.

Finally, system modeling and estimation can be done in continuous or discrete time. We will present both approaches as well as their connections. We will first introduce the basic problem of optimal estimation in discrete time along with derivations and explanations of

the cost function being minimized. We will then introduce the problem in continuous time, building off of the base provided by the introduction in discrete time. In continuous time, we will derive the estimation dynamics, present the simplification that occurs when the dynamics are linear and present convergence results before doing the same three things for discrete time.

3.1 Optimal Estimation

The optimal observer for nonlinear systems was introduced by Mortensen in 1968 in [75]. In that article, it was named the maximum likelihood estimator as the estimator was based on a minimum variance cost functional which is equivalent to a maximum likelihood cost functional.

In O. Hijab's 1980 Ph.D thesis [43], which was done under Krener's supervision, the maximum likelihood estimation technique is renamed as minimum energy estimation, this is done because the cost functional being minimized is equivalent to an energy cost functional. In recent literature, the reader will encounter this estimator being referred to as MEE or the Mortensen observer.

In discrete time, the system is:

$$\begin{cases} x_{t+1} = f(x_t) + gv_t \\ y_t = h(x_t) + kw_t \\ x_0 = x_\diamond + \chi_0 \end{cases} \quad (3.1)$$

where x_\diamond is the initial state and $\chi_0 \in \mathbb{R}^n$ is some initial disturbance. We have $v_t \in \mathbb{R}^n$ and $w_t \in \mathbb{R}^m$ which are the process and measurement disturbances. Typically $m < n$, meaning that certain state variables are not being measured. The constant matrices $g \in \mathbb{R}^{n \times n}$ and $k \in \mathbb{R}^{m \times m}$ act as weights on the disturbances. We assume that $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $h : \mathbb{R}^n \rightarrow \mathbb{R}^m$ are C^2 -mappings. We also assume that both f and h are Lipschitz continuous in \mathbb{R}^n and they satisfy linear growth conditions. Suppressing the time dependency of x_1 , and x_2 for presentation clarity, we assume:

$$\begin{aligned} |f(x_1) - f(x_2)| &\leq L|x_1 - x_2| \\ |h(x_1) - h(x_2)| &\leq L|x_1 - x_2| \\ |f(x_1)| &\leq L(1 + |x_1|) \\ |h(x_1)| &\leq L(1 + |x_1|) \end{aligned}$$

for some $L > 0$ and for all times t and all $x_1, x_2 \in \mathbb{R}^n$.

Remark 3. Note we can include a control term as an argument of the governing functions $f(\cdot)$ and $h(\cdot)$ and write $f(x_t, u_t)$ and $h(x_t, u_t)$ to estimate a control system with noise or disturbances. For our presentation, we will suppress that argument since the derivation doesn't change. The reason is that the controls $u(\cdot)$ are known and the dynamics can be re-written to take them into account, in that case time varying dynamics would be needed. \lrcorner

Remark 4. In the literature, noise and disturbance are both used in the deterministic context. In some contexts, noises is associated with the stochastic setting. In this thesis we will mainly use disturbance but even noises should be thought of as deterministic unless it is explicitly stated that we have switched to a stochastic setting. \lrcorner

Definition 8. Let $v = (v_1, v_2, \dots, v_n)^\top$, define

$$\|v\|^2 = v^\top v = v_1^2 + v_2^2 + \dots + v_n^2$$

and the weighted norm of v with respect to a positive definite matrix $M \in \mathbb{R}^{n \times n}$

$$\|v\|_M^2 = v^\top M v.$$

\lrcorner

In discrete time, the minimum energy cost functional is

$$J_t(\chi_0, v_t, w_t) = e^{\alpha t} \frac{1}{2} \|\chi_0\|_{P_0}^2 + \frac{1}{2} \sum_{s=0}^t e^{\alpha(t-s)} \left(\|v_s\|^2 + \|w_s\|^2 \right). \quad (3.2)$$

In practice, two costs are used in the discrete time case. The a-priori cost includes measurements y_s where $s = 0, 1, \dots, t-1$ but does not include the latest observation at time t , y_t . The a-posteriori cost, includes this latest measurement. The details of both and their usage will be detailed in section (3.6).

The discount rate $0 < \alpha \leq 1$ is used to set how the filter should forget, or not, past observations; it is also called a forgetting factor. A rate $\alpha = 1$ will mean that past and present observations have equal weight, a rate of $\alpha < 1$ will mean that the more far back an observation is, the more diminished its impact will be. A discount rate $\alpha < 1$ closer to 1 will do this gently while a discount rate α close to 0 will have a more noticeable effect. Note, we could also have $\alpha > 1$ in which past observations have more of an effect than new ones, but this is never done in practice. We will set $\alpha = 0$ for the rest of this work.

The objective is to estimate the state trajectory $x_t \in \mathbb{R}^n$ for $t \in \{0, T\}$ given past observations $y_s \in L^2(0, T; \mathbb{R}^m)$, $s \in \{0, t\}$. We do this by minimizing the cost functional (3.2).

We will be minimizing $J(\cdot)$ with respect to the initial disturbance χ_0 , process disturbance $v(\cdot)$ and measurement disturbance $w(\cdot)$. It may be counterintuitive to “select” disturbances to minimize the value function but in this problem formulation, they are treated as “the free variables which need to be adjusted to bring \hat{x} , [the estimate] as close as possible to x ” [19]. In [59], the author interprets this assumption in the following way: “The heuristic behind MEE is that [(3.5)] is the energy in the noise triple. Nature in her attempt to confuse us chooses the three noises in a parsimonious fashion consistent with the past controls and past measurements”. It is this interpretation of the noises that allows for optimal control techniques, notably dynamic programming, to be used in estimation.

As mentioned, Mortensen first introduced the idea of optimal estimation, with what appears at first to be a different cost functional, named the maximum likelihood functional:

$$J(T; \chi_0, v, w) = \frac{1}{2} \|x_0 - x_\diamond\|_{P_0}^2 + \frac{1}{2} \sum_{s=0}^t \left(\|x_{s+1} - f(x_s)\|_{Q^{-1}}^2 + \|y_s - h(x_s)\|_{R^{-1}}^2 \right), \quad (3.3)$$

with $P_0 \in \mathbb{R}^{n \times n}$, $Q^{-1} \in \mathbb{R}^{n \times n}$, and $R^{-1} \in \mathbb{R}^{m \times m}$ the weights on the norms on the disturbances. In the stochastic setting they are the covariance matrices of the noises. The matrices P_0 , Q^{-1} , and R^{-1} are positive semi-definite and they are also symmetric. Recall a matrix M is positive semi-definite if the real number $z^\top M z$ is non-negative for every $z \in \mathbb{R}^z$.

In Mortensen’s paper [75], the system dynamics are deterministic but the noises are assumed to be stochastic. The maximum likelihood cost functional is derived in appendix A using tools from probability.

Theorem 7. *Set $g^\top g = Q$ and $k^\top k = R$ and assume that Q and R are positive definite. The maximum likelihood cost functional (3.3) and the minimum energy cost functional (3.2) are equivalent.*

Before beginning the proof of Theorem 7, let’s introduce a useful lemma.

Lemma 2. *Let $g \in \mathbb{R}^{n \times n}$ and $k \in \mathbb{R}^{m \times m}$ and assume that $g^\top g = Q$ and $k^\top k = R$. Then the following holds: $g^\top Q^{-1} g = \mathbf{I}$ and $k^\top R^{-1} k = \mathbf{I}$.*

We will prove the lemma for $k^\top R^{-1} k$, but it is the exact same proof for $g^\top Q^{-1} g$.

Proof. Using the way k is defined:

$$k^\top R^{-1} k = k^\top (k^\top k)^{-1} k.$$

Multiply the right hand side by the identity on the left:

$$= \left((k^\top k)^{-1} (k^\top k) \right) k^\top (k^\top k)^{-1} k.$$

Move the parentheses:

$$= (k^\top k)^{-1} k^\top (k k^\top) (k^\top k)^{-1} k.$$

Since $(k k^\top) = (k^\top k)^\top$, we can rewrite it as:

$$= (k^\top k)^{-1} k^\top (k^\top k)^\top (k^\top k)^{-1} k.$$

Using the symmetry of $(k^\top k)$ and thus of R^{-1} :

$$= (k^\top k)^{-1} k^\top (k^\top k) (k^\top k)^{-1} k$$

yields:

$$\begin{aligned} (k^\top k)^{-1} k^\top k &= k^\top R^{-1} k \\ &= \mathbf{I}. \end{aligned}$$

To show that $g^\top Q^{-1} g = \mathbf{I}$ we can do the above derivation with g instead k , and with Q instead of R . \square

Now let's prove Theorem 7.

Proof. In the cost (3.2), the running cost is $\|v_s\|^2 + \|w_s\|^2$, while in (3.3), the running cost is $\|x_{s+1} - f(x_s)\|_{Q^{-1}}^2 + \|y_s - h(x_s)\|_{R^{-1}}^2$.

Now, using the system dynamics to rewrite the running cost from the maximum likelihood functional, we obtain:

$$\begin{aligned} \|x_{s+1} - f(x_s)\|_{Q^{-1}}^2 + \|y_s - h(x_s)\|_{R^{-1}}^2 &= \|g v_s\|_{Q^{-1}}^2 + \|k w_s\|_{R^{-1}}^2 \\ &= (g v_s)^\top Q^{-1} (g v_s) + (k w_s)^\top R^{-1} (k w_s) \\ &= v_s^\top g^\top Q^{-1} g v_s + w_s^\top k^\top R^{-1} k w_s. \end{aligned}$$

Using Lemma 2:

$$= v_s^\top \mathbf{I} v_s + w_s^\top \mathbf{I} w_s.$$

Thus we obtain:

$$\|x_{s+1} - f(x_s)\|_{Q^{-1}}^2 + \|y_s - h(x_s)\|_{R^{-1}}^2 = \|v_s\|^2 + \|w_s\|^2.$$

The running cost in the maximum likelihood cost functional is equivalent to the running cost in the minimum energy cost functional. \square

Now that we have introduced the cost used in MEE, we will present how the problem is solved in continuous time before introducing the approaches used in discrete time.

3.2 Minimum Energy Estimation in Continuous Time

In continuous time, a nonlinear system is:

$$\begin{cases} \dot{x}(t) = f(x(t)) + gv(t) \\ y(t) = h(x(t)) + kw(t) \\ x(0) = x_0 + \chi_0. \end{cases} \quad (3.4)$$

All the dimensions and assumptions for the state, disturbances, functions and covariances remain the same as in the discrete time formulation.

In this approach, we estimate the state trajectory of a nonlinear system by identifying the trajectory that is most consistent with the measured output while minimizing the total “energy” of the initial uncertainty, process and measurement disturbances.

Let’s introduce the continuous time energy or cost functional:

$$J(t; \chi_0, v, w) = \frac{1}{2} \|\chi_0\|_{P_0}^2 + \frac{1}{2} \int_0^t \left(\|v(s)\|^2 + \|w(s)\|^2 \right) ds. \quad (3.5)$$

In the continuous time case, the estimation problem is solved by minimizing (3.5) subject to (3.4) for all times $t \in [0, T]$.

For our presentation of MEE in continuous time, we will follow [19], [43], [58], and [75].

Before tackling the full problem of estimating every state in the trajectory by minimizing the total energy (3.5) subject to the dynamics (3.4) for all times $t \in [0, T]$, let’s break down the problem into smaller pieces and use a similar framework to that introduced in the dynamic programming section in Chapter 2. It is useful to first think of this estimation problem in the following way, minimize:

$$J(T, \xi; \chi_0, v, w) = \frac{1}{2} \|\chi_0\|_{P_0}^2 + \frac{1}{2} \int_0^T \left(\|v(s)\|^2 + \|w(s)\|^2 \right) ds. \quad (3.6)$$

subject to (3.4) with the final condition $x(T) = \xi \in \mathbb{R}^n$. We specify that ξ is some fixed point, but in the formulation of this problem, it is considered as the endpoint that minimizes the energy functional. Overall, this problem can be interpreted as finding the most likely trajectory (in terms of minimum energy) that results in the final state ξ given the observed output $y(\cdot)$ over $[0, T]$. We’ll call this problem (P^T).

We also define the equivalent of the value function, sometimes referred to as the cost-to-go:

$$V(\xi, T) = \min_{\chi_0, v, w} J(T, \xi; \chi_0, v, w).$$

Then the optimal estimate of the state at time T is:

$$\hat{x}(T) = \underset{\xi \in \mathbb{R}^n}{\operatorname{argmin}} V(\xi, T).$$

In other words, we seek the final state ξ that minimizes the energy required to generate the observations $y(\cdot)$ via the dynamic model with disturbances.

Having introduced (P^T) , a building block of the full MEE problem, we can now understand that solving the full MEE problem is equivalent to solving (P^t) , with $t \in [0, T]$ for all t . In other words, we will solve the full MEE problem by solving the collection of problems in the same form as (P^T) but for varying times t instead of only for the fixed final time T . And for each t , we define $V(\xi, t)$ as the minimal energy required for a trajectory to reach state ξ at time t while remaining consistent with the measured output.

We can rewrite (3.5) so that it doesn't depend on the initial disturbance χ_0 or measurement disturbance w :

$$J(t; v) = \frac{1}{2} \|x(0) - x_\diamond\|_{P_0}^2 + \frac{1}{2} \int_0^t \left(\|v(s)\|^2 + \|y(s) - h(x(s))\|_{R^{-1}}^2 \right) ds.$$

Notice $\|w(s)\|^2$ was rewritten as $\|y(s) - h(x(s))\|_{R^{-1}}^2$, they were shown to be equivalent in the previous section. Thus, we are now only explicitly minimizing over the process disturbance $v(\cdot)$, but both the initial condition and measurement disturbance appear indirectly through the dynamics (3.4). Accordingly, we define the equivalent of the value function, sometimes referred to as the cost-to-go:

$$V(\xi, t) = \min_v J(t, \xi; v) \tag{3.7}$$

where $\xi = x(t)$ for specific t .

Then the estimate of the state at time t is:

$$\hat{x}(t) = \underset{\xi \in \mathbb{R}^n}{\operatorname{argmin}} V(\xi, t).$$

Notice the value function here is similar to the value function in dynamic programming applied to optimal control with one key difference, $V(\xi, t)$ here minimizes the cost from time 0 to t , while a value function $V(x, t)$ in traditional dynamic programming minimizes the cost from t to T , a terminal time.

We now want to emphasize a key aspect of estimation which differs from optimal control, that is, its non-recursive nature. Let's pick times t_1, t_2 such that $0 < t_1 < t_2$. Recall the

minimum energy estimate at time t , is the endpoint of the trajectory $\hat{x}_t(\cdot)$. Consider the trajectory $\hat{x}_{t_2}(\cdot)$, its endpoint $\hat{x}_{t_2}(t_2)$ is the minimum energy estimate at time t_2 . Now consider the point $\hat{x}_{t_2}(t_1)$ of that trajectory, may not be the minimum energy estimate at time t_1 , as the trajectory $\hat{x}_{t_2}(\cdot)$ holds information computed over times $t > t_1$. The estimate $\hat{x}_{t_2}(t_1)$ exists and is known as a smoothed estimate [73].

Recall from dynamic programming and equation (2.6) that we can write down the cost-to-go at time T in the following manner:

$$V(\xi, T) = V(t, x^*(t)) + \min_{v \in L^2(t, T; \mathbb{R}^m)} \frac{1}{2} \int_t^T \left(\|v(s)\|^2 + \|y(s) - h(x(s))\|_{R^{-1}}^2 \right) ds.$$

where $x^*(t)$ is the solution to the minimization problem from $[0, t]$.

This is similar to Bellman's principle of optimality but is backwards in time in the context of estimation. The manner in which the value function is decomposed using the principle of optimality may appear to contradict the non-recursive nature of estimation, as outlined below equation (3.7). However, these two concepts can be reconciled by recognizing their distinct roles in the estimation framework.

The principle of optimality is a property of the value function that reflects the mathematical structure of the optimization problem. It asserts that the value function at a later time T can be expressed in terms of the value function at an earlier time t , augmented by the cost incurred between t and T . This decomposition is valid because the value function characterizes the minimal achievable cost-to-go from any state and time, inherently supporting a backward-in-time analysis, as required by dynamic programming.

Conversely, the non-recursive nature of estimation pertains to the computation of the state estimate at specific times. For instance, the minimum energy estimate at t_2 uses data from the entire observation window $[0, t_2]$, including information from times $t > t_1$. Consequently, the trajectory $\hat{x}_{t_2}(t)$, computed using this full dataset, generally differs from the minimum energy estimate $\hat{x}_{t_1}(t)$, which is computed using data available only up to t_1 .

This apparent conflict is resolved by distinguishing between the mathematical properties of the value function and the practical computation of the state estimate. The principle of optimality governs the structure of the optimization problem, ensuring that the minimization process is consistent and well-posed. On the other hand, the non-recursive nature of estimation arises because the process of reconstructing the system's state relies on incorporating information from the entire data set, rather than sequentially updating the state estimate in real time.

Thus, the principle of optimality remains valid and applicable to the formulation of the value function, while the non-recursive nature of estimation reflects the broader scope of

information used to compute state trajectories retrospectively. This distinction underscores the complementary roles of theoretical and practical considerations in estimation problems. Nevertheless, $V(\cdot)$ solves the HJB equation, let's derive the HJB for our specific problem. Recall from chapter 2, the control theory Hamiltonian:

$$H = p^\top \dot{x} + L(v, w)$$

where $L(v, w)$ is the running cost and p the adjoint variable. For this problem, this yields:

$$\begin{aligned} H(x, p, v, t) &= p^\top (f(x(t)) + gv(t)) - \frac{1}{2} \|v(t)\|^2 - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2 \\ &= p^\top f(x(t)) + p^\top (gv(t)) - \frac{1}{2} \|v(t)\|^2 - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2. \end{aligned} \quad (3.8)$$

Notice the negative signs on the running cost, they appear because the Hamiltonian assumes $r(v, w)$ is a running payoff for which we are trying to maximize, but in our situation we are minimizing for it, or maximizing for its negative. Applying the maximum principle, which by taking the negative of the running cost we have used for minimization purposes, we write:

$$\begin{aligned} \nabla_v H &= 0 \\ g^\top p - v &= 0 \end{aligned}$$

which then yields:

$$v = g^\top p. \quad (3.9)$$

Let's use (3.9) to substitute v in (3.8).

$$\begin{aligned} H_*(x, p, t) &= p^\top f(x(t)) + p^\top g(g^\top p) - \frac{1}{2} \|g^\top p\|^2 - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2 \\ &= p^\top f(x(t)) + p^\top (gg^\top)p - \frac{1}{2} (p^\top gg^\top p) - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2 \\ &= p^\top f(x(t)) + \frac{1}{2} (p^\top gg^\top p) - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2. \end{aligned}$$

Now write $gg^\top = Q$:

$$H_*(x, p, t) = p^\top f(x(t)) + \frac{1}{2} (p^\top Q p) - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2.$$

Rewrite using the weighted norm notation:

$$H_*(x, p, t) = p^\top f(x(t)) + \frac{1}{2} \|p\|_Q^2 - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2. \quad (3.10)$$

In Chapter 2 we introduced the HJB as being generally written in the form:

$$0 = \frac{\partial V}{\partial t}(x, t) + \inf_{u(\cdot) \in \mathcal{A}} \left\{ \nabla_x V(x, t) \cdot f(x, u) + L(t, x(t), u(t)) \right\}$$

but, as seen in that chapter, the HJB can also be re-written using the optimal Hamiltonian H_* [67]:

$$0 = \frac{\partial}{\partial t} V(x, t) + H_*(x, \nabla_x V(x, t), t).$$

Notice that $\nabla_x V(x, t)$ is in the role of the costate p .

Substituting in the optimal Hamiltonian (3.10), we obtain:

$$0 = \frac{\partial}{\partial t} V(x, t) + \nabla_x V(x, t)^\top f(x(t)) + \frac{1}{2} \|\nabla_x V(x, t)\|_Q^2 - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2.$$

We now move on to how one can find an explicit expression for the estimate $\hat{x}(t)$.

For the rest of this chapter we will use the notation $\frac{\partial}{\partial x} V$ when referring to the partial derivative of V with respect to x , instead of using $\nabla_x V$. This will make steps in the proofs that follow more clear.

3.3 Derivation of the Continuous Time Minimum Energy Estimator Dynamics

The following derivation of the estimation dynamics for the MEE in continuous time was first done by Mortensen in [75], here we follow and detail Krener's presentation from [58]. The idea is to start with the HJB equation and from there we will show that an expression for the dynamics of minimum energy estimate can be derived; it is in feedback form. Although the resulting expression is not as simple as the one found in the Kalman filter, the reader will notice certain similarities. The expression includes a term P which cannot be obtained directly. In order to obtain an equation for this term, linearity assumptions have to be made.

Because the HJB which we found above does not always yield general solutions, we use the previously introduced concept of viscosity solutions (Definition 7) to obtain to an equation for dynamics of the minimum energy estimator.

Theorem 8 (Solution of the HJB). *$V(\xi, t)$ is locally Lipschitz and solves, in a viscosity sense, the Hamilton Jacobi Bellman equation:*

$$0 = \frac{\partial}{\partial t} V(x, t) + \frac{\partial}{\partial x} V(x, t)^\top f(x(t)) + \frac{1}{2} \left\| \frac{\partial}{\partial x} V(x, t) \right\|_Q^2 - \frac{1}{2} \|y(t) - h(x(t))\|_{R^{-1}}^2 \quad (3.11)$$

where $\|x\|_A^2 = x^\top Ax$, $Q = gg^\top$, and $R = kk^\top$. The theory also states that the viscosity solution to the HJB is unique.

Furthermore, if the value function V is C^2 , its Hessian is invertible, and h is C^1 with respect to x , then the minimum energy estimate $\hat{x}(t)$ follows these dynamics:

$$\dot{\hat{x}}(t) = f(\hat{x}(t)) + P(t) \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} (y(t) - h(\hat{x}(t))),$$

where $P(t) = \left(\frac{\partial^2}{\partial x^2} V(\hat{x}(t), t) \right)^{-1}$. Here $\frac{\partial^2}{\partial x^2} V$ refers to the second derivative of V with respect to the variable x .

See [58] for this theorem with the inclusion of viscosity solutions. Recall $\hat{x}(t) = \operatorname{argmin}_\xi V(\xi, t)$ so in the general case we can use $\hat{x}(t)$ instead of the fixed point ξ .

Lemma 3. *Let V be differentiable at $(\hat{x}(t), t)$. Then*

$$0 = \frac{\partial}{\partial x} V(\hat{x}(t), t). \quad (3.12)$$

Proof. We defined $\hat{x}(t) = \operatorname{argmin}_x V(x, t)$, so by construction the value of $V(x, t)$ evaluated at $\hat{x}(t)$ i.e. $V(\hat{x}(t), t)$, is minimal so (\hat{x}, t) is a critical point of V and thus the partial derivative of V with respect to the argument x at that point is 0 [43]. \square

Lemma 4. *Assuming that both $V(\cdot)$ and $\hat{x}(\cdot)$ are differentiable. The total derivative of V evaluated at $(\hat{x}(t), t)$ is:*

$$\frac{d}{dt} V(\hat{x}(t), t) = \frac{\partial}{\partial t} V(\hat{x}(t), t). \quad (3.13)$$

Proof. By the chain rule the total derivative of V is:

$$\frac{d}{dt} V(\hat{x}(t), t) = \frac{\partial}{\partial x} V(\hat{x}(t), t) \frac{\partial}{\partial t} \hat{x}(t) + \frac{\partial}{\partial t} V(\hat{x}(t), t). \quad (3.14)$$

By Lemma 3, we know that $0 = \frac{\partial}{\partial x} V(\hat{x}(t), t)$. Evaluating (3.14) at (\hat{x}, t) and applying Lemma 3, we obtain:

$$\frac{d}{dt} V(\hat{x}(t), t) = \frac{\partial}{\partial t} V(\hat{x}(t), t). \quad \square$$

Lemma 5. *Assuming that $V \in C^2([0, T], \mathbb{R}^n)$ in a neighborhood of $(\hat{x}(t), t)$ and $\hat{x}(\cdot)$ is differentiable in a neighborhood of t , the following equation holds:*

$$0 = \frac{\partial^2}{\partial t \partial x_i} V(\hat{x}(t), t) + \sum_{j=1}^n \frac{\partial^2}{\partial x_j \partial x_i} V(\hat{x}(t), t) \dot{\hat{x}}_j(t). \quad (3.15)$$

It is important to note that $\frac{\partial}{\partial x}V(\hat{x}(t), t) = 0$, as shown in Lemma 3.12, does not imply $\frac{\partial^2}{\partial x^2}V(\hat{x}(t), t) = 0$.

Proof. Since we supposed V is $C^2([0, T], \mathbb{R}^n)$ in a neighborhood of $(\hat{x}(t), t)$ and $\hat{x}(\cdot)$ is differentiable in a neighborhood of t , we can take the total derivative with respect to time of $\frac{\partial}{\partial x}V(\hat{x}(t), t)$ and use the chain rule to obtain:

$$\frac{d}{dt}\left(\frac{\partial}{\partial x}V(\hat{x}(t), t)\right) = \frac{\partial}{\partial t}\left(\frac{\partial}{\partial x}V(\hat{x}(t), t)\right) + \frac{\partial}{\partial x}\left(\frac{\partial}{\partial x}V(\hat{x}(t), t)\right)\hat{x}(t). \quad (3.16)$$

Since:

$$\frac{\partial}{\partial x}V(\hat{x}(t), t) = \left(\frac{\partial}{\partial x_1}V(\hat{x}(t), t), \frac{\partial}{\partial x_2}V(\hat{x}(t), t), \dots, \frac{\partial}{\partial x_n}V(\hat{x}(t), t)\right)^\top.$$

Writing (3.16) in component form:

$$\frac{d}{dt}\left(\frac{\partial}{\partial x_i}V(\hat{x}(t), t)\right) = \frac{\partial}{\partial t}\left(\frac{\partial}{\partial x_i}V(\hat{x}(t), t)\right) + \sum_{j=1}^n \frac{\partial}{\partial x_j}\left(\frac{\partial}{\partial x_i}V(\hat{x}(t), t)\right)\hat{x}_j(t).$$

Applying this to (3.12):

$$0 = \frac{\partial^2}{\partial t \partial x_i}V(\hat{x}(t), t) + \sum_{j=1}^n \frac{\partial^2}{\partial x_j \partial x_i}V(\hat{x}(t), t)\dot{\hat{x}}_j(t). \quad (3.17)$$

□

Lemma 6. *The total derivative of the value function V at $(\hat{x}(t), t)$ is as follows:*

$$\frac{d}{dt}V(\hat{x}(t), t) = \frac{1}{2}\left|y(t) - h(x(t))\right|_{R^{-1}}^2. \quad (3.18)$$

Proof. Recall the HJB (3.11), is:

$$0 = \frac{\partial}{\partial t}V(x, t) + \frac{\partial}{\partial x}V(x, t)f(x(t)) + \frac{1}{2}\left|\frac{\partial}{\partial x}V(x, t)\right|_Q^2 - \frac{1}{2}\left|y(t) - h(x(t))\right|_{R^{-1}}^2.$$

Evaluating the HJB at $(\hat{x}(t), t)$ and applying Lemma 3, we obtain:

$$\frac{\partial}{\partial t}V(x, t) = \frac{1}{2}\left|y(t) - h(x(t))\right|_{R^{-1}}^2.$$

And applying Lemma 4, we obtain:

$$\frac{d}{dt}V(\hat{x}(t), t) = \frac{1}{2}\left|y(t) - h(x(t))\right|_{R^{-1}}^2. \quad (3.19)$$

□

Now we begin the proof of Theorem 8.

Proof. Recall the HJB (3.11), is:

$$0 = \frac{\partial}{\partial t} V(x, t) + \frac{\partial}{\partial x} V(x, t) f(x(t)) + \frac{1}{2} \left| \frac{\partial}{\partial x} V(x, t)^\top \right|_Q^2 - \frac{1}{2} |y(t) - h(x(t))|_{R^{-1}}^2.$$

Take the partial derivative of the HJB (3.11) with respect to x_i and write it in component form:

$$\begin{aligned} 0 = & \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial t} V(x(t), t) \right) + \sum_{j=1}^n \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial x_j} V(x(t), t) \right) f_j(x(t)) + \frac{\partial}{\partial x_i} V(x(t), t) \frac{\partial}{\partial x_i} f(x(t)) + \\ & \sum_{j=1}^n \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial x_j} V(x(t), t) \right) Q_{j,i} \frac{\partial}{\partial x_j} V(x(t), t) + \sum_{r,s} \frac{\partial}{\partial x_i} h_r(x(t)) R_{r,s}^{-1} (y_s(t) - h_r(x(t))). \end{aligned} \quad (3.20)$$

where h_r is the r^{th} component of the function h .

If we evaluate (3.20) at $(\hat{x}(t), t)$:

$$0 = \frac{\partial^2}{\partial x_i \partial t} V(\hat{x}(t), t) + \sum_{j=1}^n \left(\frac{\partial^2}{\partial x_i \partial x_j} V(\hat{x}(t), t) \right) f_j(\hat{x}(t)) + \sum_{r,s} \frac{\partial}{\partial x_i} h_r(\hat{x}(t)) R_{r,s}^{-1} (y_s(t) - h_r(\hat{x}(t))). \quad (3.21)$$

Notice, any term containing $\frac{\partial}{\partial x} V(\hat{x}(t), t)$ vanishes.

Since V is C^2 in all its arguments, partial derivatives are commutative:

$$\frac{\partial^2}{\partial x_i \partial t} V(\hat{x}(t), t) = \frac{\partial^2}{\partial t \partial x_i} V(\hat{x}(t), t),$$

We can then rewrite (3.21) as:

$$0 = \frac{\partial^2}{\partial t \partial x_i} V(\hat{x}(t), t) + \sum_{j=1}^n \left(\frac{\partial^2}{\partial x_j \partial x_i} V(\hat{x}(t), t) \right) f_i(\hat{x}(t)) + \sum_{r,s} \frac{\partial}{\partial x_i} h_r(\hat{x}(t)) R_{r,s}^{-1} (y_s(t) - h_r(\hat{x}(t))).$$

And thus by (3.17), equation (3.21) becomes:

$$\begin{aligned} 0 = & - \sum_{j=1}^n \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial x_j} V(\hat{x}(t), t) \right) \dot{\hat{x}}_j(t) + \sum_{j=1}^n \left(\frac{\partial^2}{\partial x_j \partial x_i} V(\hat{x}(t), t) \right) f_i(\hat{x}(t)) + \\ & \sum_{r,s} \frac{\partial}{\partial x_i} h_r(\hat{x}(t)) R_{r,s}^{-1} (y_s(t) - h_r(\hat{x}(t))). \end{aligned} \quad (3.22)$$

Move the first term to the left side:

$$\sum_{j=1}^n \frac{\partial^2}{\partial x_i \partial x_j} V(\hat{x}(t), t) \dot{\hat{x}}_j(t) = \sum_{j=1}^n \left(\frac{\partial^2}{\partial x_j \partial x_i} V(\hat{x}(t), t) \right) f_i(\hat{x}(t)) + \sum_{r,s} \frac{\partial}{\partial x_i} h_r(\hat{x}(t)) R_{rs}^{-1} (y_s(t) - h_r(\hat{x}(t))). \quad (3.23)$$

Now, assume that $\frac{\partial^2}{\partial x^2} V(\hat{x}(t), t)$ is invertible and define:

$$P(t) := \left(\frac{\partial^2}{\partial x^2} V(\hat{x}(t), t) \right)^{-1}.$$

We multiply (3.23) by $P(t)$ on the left and we stop showing all the components for clarity:

$$\dot{\hat{x}}(t) = f(\hat{x}(t)) + P(t) \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} (y(t) - h(\hat{x}(t))). \quad (3.24)$$

This is an ODE for $\hat{x}(t)$, our optimal estimate. Notice it is in feedback form. \square

In the next section we will make a linearity assumption and work towards obtaining some equation which can be solved to obtain $P(t)$ for such cases.

3.4 Continuous Time Minimum Energy Estimation in the Linear Case

Corollary 1. *Let f and h be twice continuously differentiable in a neighborhood of $\hat{x}(t)$. Additionally, assume that the value function V is three times continuously differentiable in a neighborhood of $(\hat{x}(t), t)$ and that the second partials of f and h and the third partials of V vanish meaning that f and h are linear. Then the covariance matrix $P(t) = \left(\frac{\partial^2}{\partial x^2} V(\hat{x}(t), t) \right)^{-1}$ satisfies the Riccati equation:*

$$\dot{P} = \frac{\partial}{\partial x} f(\hat{x}(t)) P + P \frac{\partial}{\partial x} f(\hat{x}(t))^\top + Q - P \left(\frac{\partial}{\partial x} h(\hat{x}(t)) \right)^\top R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)) P.$$

Proof. Let's take the partial derivative of the HJB (3.11) with respect to x_i then x_j . We

already took it once and obtained:

$$0 = \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial t} V(x(t), t) \right) + \sum_{j=1}^n \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial x_j} V(x(t), t) \right) f_j(x(t)) + \frac{\partial}{\partial x_i} V(x(t), t) \frac{\partial}{\partial x_i} f(x(t)) + \sum_{j=1}^n \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial x_j} V(x(t), t) \right) Q_{j,i} \frac{\partial}{\partial x_j} V(x(t), t) + \sum_{r,s} \frac{\partial}{\partial x_i} h_r(x(t)) R_{rs}^{-1} (y_s(t) - h_r(x(t))).$$

So take it one more time with respect to x_j :

$$0 = \frac{\partial}{\partial x_j} \left(\frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial t} V(x(t), t) \right) \right) + \sum_{i,j=1}^n \frac{\partial}{\partial x_j} \frac{\partial}{\partial x_i} \left(\frac{\partial}{\partial x_k} V(x(t), t) \right) f_k(x(t)) + \sum_{j=1}^n \frac{\partial}{\partial x_j} \left(\frac{\partial}{\partial x_k} V(x(t), t) \right) \frac{\partial}{\partial x_k} f(x(t)) + \frac{\partial^2}{\partial x^2} V(x(t), t) \frac{\partial}{\partial x} f(x(t)) + \frac{\partial}{\partial x} V(x(t), t) \frac{\partial^2}{\partial x^2} f(x(t)) + \frac{\partial^2}{\partial x^2} V(x(t), t) Q \frac{\partial^2}{\partial x^2} V(x(t), t) + \frac{\partial}{\partial x} V(x(t), t) Q \frac{\partial^3}{\partial x^3} V(x(t), t) + \frac{\partial^2}{\partial x^2} h(x(t)) R^{-1} (y(t) - h(x(t))) - \frac{\partial}{\partial x} h(x(t)) R^{-1} \frac{\partial}{\partial x} h(x(t)).$$

If we evaluate the above second partial derivative of at $(\hat{x}(t), t)$:

$$0 = \sum_{k=1}^n \frac{\partial}{\partial x_k} \left(\frac{\partial^2}{\partial x_i \partial x_j} V(\hat{x}(t), t) \right) f_k(\hat{x}(t)) + \sum_k \frac{\partial}{\partial x_k} \frac{\partial}{\partial x_i} V(\hat{x}(t), t) \frac{\partial}{\partial x_j} f_k(\hat{x}(t)) + \sum_k \frac{\partial}{\partial x_k} \frac{\partial}{\partial x_j} V(\hat{x}(t), t) \frac{\partial}{\partial x_i} f_k(\hat{x}(t)) + \frac{\partial^2}{\partial x_i \partial x_j} \frac{\partial}{\partial t} V(\hat{x}(t), t) + \sum_{k,l} \frac{\partial}{\partial x_k} V(\hat{x}(t), t) Q_{k,l} \frac{\partial}{\partial x_l} \frac{\partial}{\partial x_j} V(\hat{x}(t), t) + \frac{\partial^2}{\partial x^2} h(\hat{x}(t)) R^{-1} (y(t) - h(\hat{x}(t))) - \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)). \quad (3.25)$$

Again, the terms containing $\frac{\partial}{\partial x} V(\hat{x}(t), t)$ vanish.

We now set the second partials of f and h and the third partials of V to 0; obtaining:

$$\begin{aligned}
0 = & \frac{\partial^2}{\partial x_i \partial x_j} \frac{\partial}{\partial t} V(\hat{x}(t), t) + \\
& \sum_k \frac{\partial}{\partial x_k} \frac{\partial}{\partial x_i} V(\hat{x}(t), t) \frac{\partial}{\partial x_j} f_k(\hat{x}(t)) + \sum_k \frac{\partial}{\partial x_k} \frac{\partial}{\partial x_j} V(\hat{x}(t), t) \frac{\partial}{\partial x_i} f_k(\hat{x}(t)) + \\
& \sum_{k,l} \frac{\partial}{\partial x_k} V(\hat{x}(t), t) Q_{k,l} \frac{\partial}{\partial x_l} \frac{\partial}{\partial x_j} V(\hat{x}(t), t) - \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)).
\end{aligned} \tag{3.26}$$

Recall

$$P(t) = \left(\sum_k \frac{\partial}{\partial x_k} \frac{\partial}{\partial x_i} V(\hat{x}(t), t) \right)^{-1},$$

so

$$P(t)^{-1} = \frac{\partial^2}{\partial x^2} \sum_k \frac{\partial}{\partial x_k} \frac{\partial}{\partial x_i} V(\hat{x}(t), t). \tag{3.27}$$

We now use the commutativity of partials and (3.27) to re-write (3.26). We drop the time dependency of P for clarity,

$$\begin{aligned}
0 = & \frac{\partial}{\partial t} P^{-1} + P^{-1} \frac{\partial}{\partial x_j} f(\hat{x}(t)) + P^{-1} \frac{\partial}{\partial x_i} f(\hat{x}(t)) + \\
& P^{-1} Q P^{-1} - \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)).
\end{aligned}$$

Left multiply by P :

$$\begin{aligned}
0 = & P \frac{\partial}{\partial t} P^{-1} + P \cdot P^{-1} \frac{\partial}{\partial x_j} f(\hat{x}(t)) + P \cdot P^{-1} \frac{\partial}{\partial x_i} f(\hat{x}(t)) \\
& P \cdot P^{-1} Q P^{-1} - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)) \\
0 = & P \frac{\partial}{\partial t} P^{-1} + \frac{\partial}{\partial x_j} f(\hat{x}(t)) + \frac{\partial}{\partial x_i} f(\hat{x}(t)) + Q P^{-1} - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)).
\end{aligned} \tag{3.28}$$

We have:

$$\frac{\partial}{\partial t} P^{-1} = -P^{-1} \cdot \dot{P} \cdot P^{-1}.$$

Substitute that into (3.28)

$$\begin{aligned}
0 = & P(-P^{-1} \cdot \dot{P} \cdot P^{-1}) + \frac{\partial}{\partial x_j} f(\hat{x}(t)) + \frac{\partial}{\partial x_i} f(\hat{x}(t)) + \\
& Q P^{-1} - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)).
\end{aligned}$$

Simplify:

$$0 = -\dot{P} \cdot P^{-1} + \frac{\partial}{\partial x_j} f(\hat{x}(t)) + \frac{\partial}{\partial x_i} f(\hat{x}(t)) + \\ QP^{-1}P - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)).$$

And we now right multiply by P:

$$0 \cdot P = (-\dot{P} \cdot P^{-1})P + \frac{\partial}{\partial x_j} f(\hat{x}(t))P + \frac{\partial}{\partial x_i} f(\hat{x}(t))P + \\ QP^{-1}P - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t))P.$$

Simplifying:

$$\dot{P} = \frac{\partial}{\partial x} f(\hat{x}(t))P + P \frac{\partial}{\partial x} f(\hat{x}(t))^\top + Q - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t))P. \quad (3.29)$$

When solved, the above equation yields P which can be used in (3.24) to obtain our estimate. \square

When solved, the above equation yields P which can be used in (3.24) to obtain our estimate. In [19], the assumption that

$$\frac{\partial^3}{\partial x_k \partial x_j \partial x_i} V(\hat{x}(t), t) = 0$$

is not made and thus the resulting estimation form in that case is

$$\dot{\hat{x}}(t) = f(\hat{x}(t)) + P(t) \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} (y(t) - h(\hat{x}(t))),$$

where

$$P(t) := \left(\frac{\partial^2}{\partial x^2} V(\hat{x}(t), t) \right)^{-1}$$

is the solution of

$$\dot{P} = \frac{\partial}{\partial x} f(\hat{x}(t))P + P \frac{\partial}{\partial x} f(\hat{x}(t))^\top + Q - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t))P + \\ P \left(\sum_{k=1}^n \frac{\partial}{\partial x_k} \frac{\partial^2}{\partial x_i \partial x_j} V(\hat{x}(t), t) f_k(\hat{x}(t)) \right) P.$$

Let's introduce the linear system:

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t) + gv(t) \\ y(t) &= Cx(t) + kw(t),\end{aligned}\tag{3.30}$$

with $x(t) \in \mathbb{R}^n$ the system state and $u(t) \in \mathbb{R}^m$ the control input.

Assume $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, and $C \in \mathbb{R}^{l \times n}$ are constant mappings. We also have the process disturbance $v(t) \in \mathbb{R}^n$ and measurement disturbance $w(t) \in \mathbb{R}^l$. We also have $gg^\top = Q \in \mathbb{R}^{n \times n}$ and $kk^\top = R \in \mathbb{R}^{l \times l}$, where $g \in \mathbb{R}^{n \times n}$ and $k \in \mathbb{R}^{l \times l}$. We'll also assume the initial state $x(0) = x_\diamond$ is known.

Remark 5. Since the controls $u(\cdot)$ are assumed to be known, we can write them in the usual feedback form:

$$u(t) = F(t)x(t).$$

The system then becomes:

$$\begin{cases} \dot{x}(t) = Ax(t) + B \cdot Fx(t) + v(t) \\ y(t) = Cx(t) + w(t). \end{cases}$$

We can then define $\tilde{A} = A + B \cdot F$ and rewrite the dynamics as:

$$\begin{cases} \dot{x}(t) = \tilde{A}x(t) + v(t) \\ y(t) = Cx(t) + w(t). \end{cases}\tag{3.31}$$

We will use the system (3.30) for this section but the simplification would be the same with the system (3.31). ┘

Theorem 9. *If the MEE is applied to the linear system (3.30), it is equivalent to the Kalman filter. For more background on the Kalman filter see [51] and [95].*

Proof. Recall the MEE equations (3.24) and (3.29):

$$\dot{\hat{x}}(t) = f(\hat{x}(t)) + P(t) \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} (y(t) - h(\hat{x}(t)))\tag{3.32}$$

and

$$\dot{P} = \frac{\partial}{\partial x} f(\hat{x}(t)) P + P \frac{\partial}{\partial x} f(\hat{x}(t))^\top + Q - P \frac{\partial}{\partial x} h(\hat{x}(t)) R^{-1} \frac{\partial}{\partial x} h(\hat{x}(t)) P.\tag{3.33}$$

So we have:

$$f(x(t), u(t)) = Ax(t) + Bu(t) + v(t)$$

$$h(x(t), u(t)) = Cx(t) + w(t).$$

(3.32) becomes:

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + v + P(t)CR^{-1}(y(t) - (C\hat{x}(t) + w(t)))$$

since

$$\frac{\partial}{\partial x}h(\hat{x}) = C.$$

And (3.33) becomes:

$$\dot{P} = AP + PA^\top + Q - PCR^{-1}CP. \quad (3.34)$$

In the Kalman filter formulation, we usually write:

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + v + K(y - C\hat{x}) \\ K &= PC^\top R^{-1}, \end{aligned}$$

where P is the solution to the following differential Riccati equation (3.34). K is known as the Kalman gain and is the piece of the equation which incorporates new measurements into the known dynamics to obtain an optimal estimate.

We were indeed able to retrieve the Kalman filter equations by using the MEE equations for a linear system. \square

Remark 6. Since the Riccati equation does not depend on measurements $y(\cdot)$, it can be computed offline. \lrcorner

Remark 7. The Kalman filter is often derived ([51], [69]) in the context of a dynamic system with stochastic noises but as shown above, it still appears under deterministic assumptions. \lrcorner

3.5 Convergence of the Continuous Time Minimum Energy Estimator

Before introducing the existing results for convergence of the MEE in continuous time, we will present some important definitions related to dynamical systems, the notions of observability and the weaker concept of detectability. These will be used in the conditions for convergence.

Kalman introduced observability in [52] and [53]. For linear systems, there is duality between controllability in control theory and observability in estimation theory.

Definition 9 (Continuous Time Observability). [Definition 7, Chapter 1 in [95]] A continuous-time system is *observable* if for any initial state x_0 and any final time $t > 0$ the initial state x_0 can be uniquely determined by knowing the input $u(\tau)$ and output $y(\tau)$ for any time $\tau \in [0, t]$. \lrcorner

Observability for linear systems is simple to check. The system

$$\begin{cases} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{cases}$$

is observable if the observability matrix Q defined as:

$$Q = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

is of full rank. To better understand the conditions of observability for nonlinear systems that we will present, we will quickly derive and explain the above condition. A key aspect of the definition of observability given above is uniqueness. A different but equivalent definition of observability is that for every pair of initial conditions x_{0_1}, x_{0_2} the corresponding outputs $y_1(t), y_2(t)$ should differ at some point $t \geq 0$ [54]. Using the rank of the observability matrix Q as a condition comes from taking the $n - 1^{\text{th}}$ time derivatives of y :

$$\begin{aligned} y(0) &= Cx_0 \\ \dot{y}(0) &= CAx_0 \\ \ddot{y}(0) &= CA^2x_0 \\ &\vdots \\ y^{(n-1)}(0) &= CA^{n-1}x_0. \end{aligned} \tag{3.35}$$

The system is observable if the mapping of x_0 to the derivatives of $y(t)$ is one-to-one i.e. that the observability matrix Q has full rank [54]. There are a multitude of other ways that observability for linear systems can be checked, see [Section 1.7.2 in [95]] for more details.

For nonlinear systems, the question of observability is more complicated. Taking the same

idea as in (3.35) we obtain:

$$\begin{aligned}
y(0) &= y(x_0) \\
\dot{y}(0) &= L_f(h)(x_0) \\
\ddot{y}(0) &= L_f^2(h)(x_0) \\
&\vdots \\
y^{(k-1)}(0) &= L_f^{k-1}(h)(x_0)
\end{aligned} \tag{3.36}$$

where

$$L_f(h)(x_0) = \frac{\partial h}{\partial x}(x_0)f(x_0)$$

and

$$L_f^2(h)(x_0) = \frac{\partial L_f(h)}{\partial x}(x_0)f(x_0)$$

are Lie derivatives. If the mapping from x_0 to $h, L_f(h), L_f^2(h), \dots$ distinguishes points (similarly to the condition for linear systems) then the system is observable. We can also check if the matrix:

$$\begin{bmatrix}
\frac{\partial h}{\partial x}(x_0) \\
\frac{\partial L_f(h)}{\partial x}(x_0) \\
\vdots \\
\frac{\partial L_f^{n-1}(h)(x_0)}{\partial x}(x_0)
\end{bmatrix}$$

is invertible which would indicate local observability at x_0 . For nonlinear systems, a differential approach to checking for observability is introduced in [42]. It evaluates observability by computing the Lie derivatives around an initial point. In practice, however, Lie derivatives are often avoided due to their high computational cost and the requirement to compute higher order derivatives. Furthermore, the rank condition which emerges is qualitative and hard to optimize [60]. A thorough treatment of observability of nonlinear systems is given in [42], including breaking the notion of an observable system down to weaker conditions such as locally observable, weakly observable and locally weakly observable. Other methods in the literature include an observation Gramian and a moving horizon approach. Observability for nonlinear systems is an active area of research.

We now introduce the notion uniform observability, it uses an alternative variable system. Two examples have been added below the definition for clarity.

Definition 10. [58] The nonlinear system:

$$\dot{x}(t) = f(x(t), u(t)) + gv(t) \tag{3.37}$$

$$y(t) = h(x(t), u(t)) + kw(t)$$

is uniformly observable for any input if there exist coordinates

$$\{\tilde{x}_{ij} : i = 1, \dots, p, \quad j = 1, \dots, l_i\}$$

(i is the index keeping track of the different observable blocks) where $1 \leq l_1 \leq \dots \leq l_p$ and $\sum l_i = n$ such that in these coordinates, the system takes the observable canonical form:

$$\begin{aligned} y_i &= \tilde{x}_{i1} + h_i(u) \\ \dot{\tilde{x}}_{i1} &= \tilde{x}_{i2} + f_{i1}(\underline{x}_1, u) + g_{i1}(\underline{x}_1)v \\ &\vdots \\ \dot{\tilde{x}}_{ij} &= \tilde{x}_{ij+1} + f_{ij}(\underline{x}_j, u) + g_{ij}(\underline{x}_j)v \\ &\vdots \\ \dot{\tilde{x}}_{il_{i-1}} &= \tilde{x}_{il_i} + f_{il_{i-1}}(\underline{x}_{l_{i-1}}, u) + g_{il_{i-1}}(\underline{x}_{l_{i-1}})v \\ \dot{\tilde{x}}_{il_i} &= f_{il_i}(\underline{x}_{l_i}, u) + g_{il_i}(\underline{x}_{l_i})v \end{aligned}$$

for $i = 1, \dots, p$ and where \underline{x}_j has the following definition:

$$\underline{x}_j := (\tilde{x}_{11}, \dots, \tilde{x}_{1j \wedge l_1}, \tilde{x}_{21}, \dots, \tilde{x}_{pj}).$$

The definition of \underline{x}_j implies that \underline{x}_j is the collection of points

The indices i range over $i = 1, \dots, p$ and the second index k over $k = 1, \dots, j \wedge l_i$ where $a \wedge b = \min\{a, b\}$. The coordinates are ordered in such a way that the right index k moves faster than the left index i .

In the simplest case where $p = 1$, there is only one observable block and \underline{x}_j is just the collection of all variables before \tilde{x}_{1j} and including \tilde{x}_{1j} , so $\underline{x}_j = \{\tilde{x}_{11}, \tilde{x}_{12}, \dots, \tilde{x}_{1j}\}$ in this case.

Now consider a case where there are 3 blocks. Recall l_i is the number of states in block i . In this example let us have $l_1 = 3$, $l_2 = 2$, $l_3 = 4$. Then we would have:

$$\begin{aligned} \underline{x}_1 &= (\tilde{x}_{11}, \tilde{x}_{21}, \tilde{x}_{31}) \\ \underline{x}_2 &= (\tilde{x}_{11}, \tilde{x}_{12}, \tilde{x}_{21}, \tilde{x}_{22}, \tilde{x}_{31}, \tilde{x}_{32}) \\ \underline{x}_3 &= (\tilde{x}_{11}, \tilde{x}_{12}, \tilde{x}_{13}, \tilde{x}_{21}, \tilde{x}_{22}, \tilde{x}_{31}, \tilde{x}_{32}, \tilde{x}_{33}) \\ \underline{x}_4 &= (\tilde{x}_{11}, \tilde{x}_{12}, \tilde{x}_{13}, \tilde{x}_{21}, \tilde{x}_{22}, \tilde{x}_{31}, \tilde{x}_{32}, \tilde{x}_{33}, \tilde{x}_{34}) \end{aligned}$$

The above example illustrates how the definition of \underline{x}_j collects all terms of any block up to index j , or the maximum index in that block if j exceeds the number of states in a block.

In the systems we are working with this condition is already required, we mention it anyway. Recall that each f_{ij}, g_{ij} is Lipschitz continuous and satisfies growth conditions as follows:

$$\begin{aligned} |f_{ij}(\underline{x}_j, u) - f_{ij}(\underline{z}_j, u)| &\leq L|\underline{x}_j - \underline{z}_j| \\ |g_{ij}(\underline{x}_j) - g_{ij}(\underline{z}_j)| &\leq L|\underline{x}_j - \underline{z}_j| \\ |f_{ij}(\underline{x}_j, u)| &\leq (L + 1)|\underline{x}_j| \\ |g_{ij}(\underline{x}_j)| &\leq L. \end{aligned}$$

The Lipschitz continuity of f_{ij} and g_{ij} ensures that small perturbations in x do not lead to arbitrarily large deviations in the system's evolution, which is a necessary condition for observability analysis. \lrcorner

To illustrate how the observable canonical form works in practice, we consider a simple system where we differentiate the observed output to reconstruct all state variables.

Example 1.

$$\begin{aligned} \dot{x}_1 &= x_2 + u \\ \dot{x}_2 &= x_3 \\ \dot{x}_3 &= \cos(x_1) + x_2 \\ y &= x_1. \end{aligned}$$

We notice the only measured state is x_1 ; however, the other states can be recovered in the following manner:

We know $y = x_1$ so $\dot{y} = \dot{x}_1$ and $\dot{x}_1 = x_2 + u$. By differentiating the observed variable we were able to recover x_2 . Let's differentiate y once more, $\ddot{y} = \ddot{x}_1$ and $\ddot{x}_1 = \dot{x}_2$, which, from the dynamics, yields $\dot{x}_2 = x_3$. We have thus recovered all the system states by differentiating the observed variable.

We can set up a change of variables:

$$\begin{aligned} y &= \tilde{x}_{11} \\ \tilde{x}_{11} &= x_1 \\ \tilde{x}_{12} &= x_2 \\ \tilde{x}_{13} &= x_3. \end{aligned}$$

Write the dynamical system in the following way:

$$\begin{aligned}\dot{\tilde{x}}_{11} &= \tilde{x}_{12} + u \\ \dot{\tilde{x}}_{12} &= \tilde{x}_{13} \\ \dot{\tilde{x}}_{13} &= \cos(\tilde{x}_{11}) + \tilde{x}_{12} \\ y &= \tilde{x}_{11}.\end{aligned}$$

In a more general sense, the system can be rewritten as:

$$\begin{aligned}y &= \tilde{x}_{11} \\ \dot{\tilde{x}}_{11} &= \tilde{x}_{12} + f_{11}(\underline{x}_1) \\ \dot{\tilde{x}}_{12} &= \tilde{x}_{13} + f_{12}(\underline{x}_2) \\ \dot{\tilde{x}}_{13} &= f_{13}(\underline{x}_3),\end{aligned}$$

where \underline{x}_j is the collection of all prior state variables. So if we only have one observable block, we would simply write $\underline{x}_j = \{x_{11}, x_{12}, \dots, x_{1j}\}$. This is a simplification of the more general definition of \underline{x}_j given in Definition 10.

In the state variable \tilde{x}_{ij} , the i refers to the observable block. In certain cases, as in the next example, not all state variables can be recovered from a single observed variables, in those cases there are more than one observable block i . The index j refers to the different variables within an observable block.

Let's give one more example to show the method in a system with more complexity.

Example 2. Consider the nonlinear system:

$$\begin{aligned}\dot{x}_1 &= x_2 + u_1 \\ \dot{x}_2 &= x_3 + \sin(x_1) \\ \dot{x}_3 &= \cos(x_2) + x_4 \\ \dot{x}_4 &= \tan(x_3) \\ \dot{x}_5 &= x_6 + u_2 \\ \dot{x}_6 &= x_7 + x_5^2 \\ \dot{x}_7 &= x_8 + \cos(x_6) \\ \dot{x}_8 &= \sin(x_7) \\ y_1 &= x_1 \\ y_2 &= x_5.\end{aligned}$$

The observed variable y_1 gives access to the first subsystem with state variables (x_1, x_2, x_3, x_4) and y_2 gives access to the first subsystem with state variables (x_5, x_6, x_7, x_8) . We will not show the step by step differentiation to recover the state variables as we had done earlier but rather we will directly re-write the system in the new variables:

$$\begin{array}{l|l}
 y_1 = \tilde{x}_{11} & y_2 = \tilde{x}_{21} \\
 \dot{\tilde{x}}_{11} = \tilde{x}_{12} + u_1 & \dot{\tilde{x}}_{21} = \tilde{x}_{22} + u_2 \\
 \dot{\tilde{x}}_{12} = \tilde{x}_{13} + \sin(\tilde{x}_{11}) & \dot{\tilde{x}}_{22} = \tilde{x}_{23} + \tilde{x}_{21}^2 \\
 \dot{\tilde{x}}_{13} = \cos(\tilde{x}_{12}) + \tilde{x}_{14} & \dot{\tilde{x}}_{23} = \tilde{x}_{24} + \cos(\tilde{x}_{22}) \\
 \dot{\tilde{x}}_{14} = \tan(\tilde{x}_{13}). & \dot{\tilde{x}}_{24} = \sin(\tilde{x}_{23}).
 \end{array}$$

Subsystem 1
Subsystem 2

More generally, we can write the system as follows:

$$\begin{array}{l|l}
 y_1 = \tilde{x}_{11} & y_2 = \tilde{x}_{21} \\
 \dot{\tilde{x}}_{11} = \tilde{x}_{12} + f_{11}(\underline{x}_1) & \dot{\tilde{x}}_{21} = \tilde{x}_{22} + f_{21}(\underline{x}_1) \\
 \dot{\tilde{x}}_{12} = \tilde{x}_{13} + f_{12}(\underline{x}_2) & \dot{\tilde{x}}_{22} = \tilde{x}_{23} + f_{22}(\underline{x}_2) \\
 \dot{\tilde{x}}_{13} = \tilde{x}_{14} + f_{13}(\underline{x}_3) & \dot{\tilde{x}}_{23} = \tilde{x}_{24} + f_{23}(\underline{x}_3) \\
 \dot{\tilde{x}}_{14} = f_{14}(\underline{x}_4). & \dot{\tilde{x}}_{24} = f_{24}(\underline{x}_4).
 \end{array}$$

Subsystem 1
Subsystem 2

These examples illustrate how Definition 10 translates into practical applications. The first example demonstrates the step-by-step differentiation process that reveals hidden states, while the second example highlights how multiple observable subsystems can be analyzed separately in the structured canonical form.

The following lemma allows us to write the canonical observable form in a more compact manner.

Lemma 7. [58] *Suppose the system (3.37) is uniformly observable for any input, then without loss of generality, it can be written in the form:*

$$\begin{aligned}
 \dot{x} &= Ax + \bar{f}(x, u) + \bar{g}(x)v \\
 y &= Cx + \bar{h}(u) + k(x)w
 \end{aligned}$$

where:

$$A = \begin{bmatrix} A_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & A_p \end{bmatrix}^{n \times n} \quad \text{with } A_i = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ & & & \ddots & \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}^{l_i \times l_i} \quad (3.38)$$

$$C = \begin{bmatrix} C_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & C_p \end{bmatrix}^{p \times n} \quad \text{with } C_i = [1 \ 0 \ 0 \ \dots \ 0]^{1 \times l_i} \quad (3.39)$$

$$\bar{f}(x, u) = \begin{bmatrix} \bar{f}_1(x, u) \\ \vdots \\ \bar{f}_p(x, u) \end{bmatrix}^{n \times 1} \quad \text{with } \bar{f}_i(x, u) = \begin{bmatrix} f_{i1}(\underline{x}_1, u) \\ \vdots \\ f_{i,l_i}(\underline{x}_{l_i}, u) \end{bmatrix}^{l_i \times 1} \quad (3.40)$$

$$\bar{g}(x, u) = \begin{bmatrix} \bar{g}_1(x, u) \\ \vdots \\ \bar{g}_p(x, u) \end{bmatrix}^{n \times l} \quad \text{with } \bar{g}_i(x, u) = \begin{bmatrix} g_{i1}(\underline{x}_1, u) \\ \vdots \\ g_{i,l_i}(\underline{x}_{l_i}, u) \end{bmatrix}^{l_i \times l}$$

$$\bar{h}(u) = [h_1(u), \dots, h_p(u)]. \quad (3.41)$$

See page 201 in [58] for more details and the proof.

With these notions in place, let's return to the existing results for convergence of the minimum energy estimator.

We are now ready to introduce a theorem for the convergence of the MEE.

Theorem 10. (Theorem 3 in [58]) *Suppose the nonlinear system (3.37) is uniformly observable, then Lemma 7 applies. We also have that $x(t), u(t), y(t)$ are any state, control output trajectories generated by the system. Let $V(\xi, t)$ be defined as in (3.7) and $\hat{x}(t)$ is piecewise continuous trajectory minimizing $V(\xi, t)$. Then $|x(t) - \hat{x}(t)| \rightarrow 0$ as $t \rightarrow \infty$.*

3.6 Minimum Energy Estimation in Discrete Time

We now present the MEE in discrete time. In 2018, Moireau in [73] used the same dynamic programming strategy as is used in the continuous MEE counterpart to find a series

of expressions for the discrete time minimum energy estimator. A discrete-time filter is said to be a stable discretization of its continuous-time counterpart if it satisfies two main properties. First, the discretized scheme must be consistent, meaning that as the time step $\Delta t \rightarrow 0$, its solution converges to the continuous-time solution given by the Mortensen observer. Second, the scheme must be stable, meaning that errors, from numerical approximations or perturbations in the measurements, do not grow unboundedly over the discrete steps. The MEE is both optimal for nonlinear systems and is a stable discretization of the equations found in the continuous time setting MEE presented above.

We will work with the non-linear discrete dynamics:

$$\begin{aligned}x_{n+1} &= f(x_n) + gv_n \\y_n &= h(x_n) + kw_n \\x_0 &= x_\diamond + \chi_0.\end{aligned}\tag{3.42}$$

with the assumptions outlined for equation (3.1) on page 25.

Recall the discrete energy cost (3.2) on page 26, we will re-write it using $\|w_s\|^2 = \|y_s - h(x_s)\|_{R^{-1}}^2$ and we will split it up into the a-priori cost J_n^- which includes system states and measurements up to time $n - 1$ but not the measurement at time n .

For $n > 0$

$$J_n^-(\chi_0, v_k) = \frac{1}{2}\|x_0 - x_\diamond\|_{P_0} + \frac{1}{2}\sum_{s=0}^{n-1}\|v_s\|^2 + \frac{1}{2}\sum_{s=0}^{n-1}\|y_s - h(x_s)\|_{R^{-1}}^2$$

and when $n = 0$:

$$J_0^-(\chi_0) = \frac{1}{2}\|x_0 - x_\diamond\|_{P_0} + \frac{1}{2}\|y_0 - h(x_0)\|_{R^{-1}}^2.$$

We also have the a-posteriori cost J_n^+ in which the latest measurement at time n is included:

$$J_n^+(\chi_0, v_k) = \frac{1}{2}\|x_0 - x_\diamond\|_{P_0} + \frac{1}{2}\sum_{s=0}^{n-1}\|v_s\|^2 + \frac{1}{2}\sum_{s=0}^n\|y_s - h(x_s)\|_{R^{-1}}^2$$

with:

$$J_0^+(\chi_0) = \frac{1}{2}\|x_0 - x_\diamond\|_{P_0} + \frac{1}{2}\|y_0 - h(x_0)\|_{R^{-1}}^2.$$

The matrices P_0 and R^{-1} are positive definite and defined in the same way as in section (3.1).

We set:

$$(\chi_{0|n}^+, v_{k|n}^+) = \underset{\chi_0, v_k}{\operatorname{argmin}} J_n^+(\chi_0, v_k) \quad (3.43)$$

and

$$(\chi_{0|n}^-, v_{k|n}^-) = \underset{\chi_0, v_k}{\operatorname{argmin}} J_{n+1}^-(\chi_0, v_k). \quad (3.44)$$

These minimization problems have associated trajectories that characterize the best possible reconstruction of the system's behavior given the available measurements.

The a-posteriori trajectory, denoted by $\hat{x}_{k|n}^+$, represents the optimal state estimate at time k incorporating all measurements up to time n . It is obtained by solving the optimization problem (3.43) and is defined as:

$$\hat{x}_{k|n}^+ = x_{k|\chi_0^+, v_{k|n}^+}.$$

Similarly, the a-posteriori trajectory, denoted by $\hat{x}_{k|n}^-$, represents the optimal state estimate at time k incorporating all measurements up to time n . It is obtained by solving the optimization problem (3.44) and is defined as:

$$\hat{x}_{k|n}^- = x_{k|\chi_0^-, v_{k|n}^-}.$$

The trajectories $\hat{x}_{k|n}^+$ and $\hat{x}_{k|n}^-$ evolve recursively according to the system dynamics:

$$\begin{aligned} \hat{x}_{0|n}^+ &= x_\diamond + \chi_0^+|_n \\ \hat{x}_{k+1|n}^+ &= f(\hat{x}_{k|n}^+) + g v_{k|n}^+ \end{aligned}$$

and

$$\begin{aligned} \hat{x}_{0|n}^- &= x_\diamond + \chi_0^-|_n, \\ \hat{x}_{k+1|n}^- &= f(\hat{x}_{k|n}^-) + g v_{k|n}^-. \end{aligned}$$

Here, $\hat{x}_{0|n}^+$ and $\hat{x}_{0|n}^-$ represent the initial conditions for the a-posteriori and a-priori state estimates, respectively. The evolution equations describe the propagation of these estimates using the optimal disturbance sequences $v_{k|n}^+$ and $v_{k|n}^-$, which are determined from the respective cost minimizations.

We introduce the prediction cost-to-come:

$$V_n^+ = \min_{(v_k)_{k < n}, \chi_0} J_n^+(\chi_0, v_k)$$

and correction cost-to-come:

$$V_n^- = \min_{(v_k)_{k \leq n}, \chi_0} J_n^-(\chi_0, v_k).$$

Theorem 11. [73] (Theorem 2.6) Assuming that for all n , V_n^- and V_n^+ are $C^2(\mathbb{R}^n)$, then V_n^- and V_n^+ are solutions to the following system :

$$\begin{cases} V_0^-(x) = \frac{1}{2} \|x - \bar{x}_0\|_{P_0}^2 \\ V_n^+(x) = V_n^-(x) + \frac{1}{2} \|y_n - h_n(x_n)\|_{R^{-1}}^2 \\ V_{n+1}^-(x) = V_n^+(\hat{x}_n^+) + \frac{1}{2} \nabla V_{n+1}^-(x)^\top g Q g^\top \nabla V_{n+1}^-(x) \\ \text{with } x = f(\hat{x}_n^+) + g Q g^\top \nabla V_{n-1}^-(x). \end{cases}$$

The above prediction-correction scheme can be used computationally with the following algorithm.

Initialization:

$$\begin{aligned} \hat{x}_0 &= \bar{x}_0 \\ V_0^- &= \frac{1}{2} \|x - \bar{x}_0\|_{P_0}^2 \\ &= 0. \end{aligned}$$

Correction:

$$\nabla V_n^+(\hat{x}_n^+) = 0.$$

Prediction:

$$x_{n+1}^- = f(\hat{x}_n^+).$$

Notice that \hat{x}_n^+ is defined implicitly as the minimizer of the cost function V_n^+ , meaning it satisfies the first order optimality condition:

$$\nabla V_n^+(\hat{x}_n^+) = 0.$$

Unlike explicit formulas that directly express a variable in terms of known quantities, this equation defines \hat{x}_n^+ indirectly; determining \hat{x}_n^+ requires an equation to be solved instead of

direct substitution. In the case where the Hessian $\nabla^2 V_n^+$ is invertible, a Newton-Raphson [7] procedure can be applied and \hat{x}_n^+ computed as the limit of the following procedure.

$$\begin{aligned}\hat{x}_{0|n}^+ &= \hat{x}_n, \quad n \in \mathbb{N} \\ \hat{x}_{k+1|n}^+ &= \hat{x}_{k|n}^+ - (\nabla^2 V_n^+(\hat{x}_{k|n}^+))^{-1} \nabla V_n^+(\hat{x}_{k|n}^+), \quad k \in \mathbb{N}.\end{aligned}$$

Similarly to the continuous setting, the equations for the MEE depend upon the inverse of the Hessian of the value function, or the cost-to-come.

The above procedure, after one iteration, can be written as:

$$\hat{x}_{k+1|n}^+ = \hat{x}_n^- - (\nabla^2 V_n^+(\hat{x}_n^-))^{-1} \frac{\partial}{\partial x} (y_n - h(\hat{x}_n^-))^\top R^{-1} (y_n - h(\hat{x}_n^-)).$$

There are other approaches to non-linear estimation which use approximations, or linearize the dynamics in order to render computations more doable. But before presenting them we will show how the above discrete time MEE formulation reduces to the Kalman filter when the system is linear.

3.7 Discrete Time Linear Case

A detailed version of the following can be found in [Proposition 2.8 in [73]] proved in [Section 4.3] In discrete time, we write the system as:

$$\begin{aligned}x_{n+1} &= Ax_n + gv_n \\ y_n &= Cx_n + kw_n.\end{aligned}$$

Again, $v_n \in \mathbb{R}^n$ and $w_n \in \mathbb{R}^l$ are white, zero-mean, uncorrelated disturbances with respective covariance matrices $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{l \times l}$. For all n , the mappings $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, and $C_n \in \mathbb{R}^{l \times n}$ are linear and the input $u_n \in \mathbb{R}^m$ is known and $x_n \in \mathbb{R}^n$ is the state with initial condition at time $n = 0$, $x_0 = x_\diamond$.

We initialize:

$$\begin{aligned}\hat{x}_0^+ &= \mathbb{E}(x_\diamond) \\ P_0^+ &= \mathbb{E}[(x_\diamond - \hat{x}_0^+)(x_\diamond - \hat{x}_0^+)^\top]\end{aligned}$$

where $\mathbb{E}(x)$ is the expected value of x .

For each $n = 1, 2, 3, \dots$ we compute the prediction using the following equations:

$$\hat{x}_n^- = A_{n-1} \hat{x}_{n-1}$$

$$P_n^- = A_{n-1}P_{n-1}^+A_{n-1}^\top + Q_{n-1}.$$

And finally, the update equations:

$$\begin{aligned} K_n &= P_n^- H_n^\top (H_n P_n^- H_n^\top + R_n)^{-1} \\ x_n^+ &= \hat{x}_n^- + K_n (y_n - H_n \hat{x}_n^-) \\ P_n^+ &= (I - K_n H_n) P_n^- (I - K_n H_n)^\top + K_n R_n K_n^\top. \end{aligned}$$

The equations for K_n and P_n^+ can be written in different ways, for an extensive and interesting discussion on this topic see chapter 5 in [95].

Remark 8. Notice in the update step, x_n^- is computed without knowledge of the measurement at time n , it is known as the a-priori estimate. In the correction step, the measurement becomes available and is taken into account for the computation of the a-posteriori estimate x_n^+ . This order of computation also occurs with P_n^- the a-priori covariance and P_n^+ the a-posteriori covariance. ┘

3.8 Relating Discrete and Continuous-Time Filters

The continuous-time Kalman filter can be derived from the discrete-time version, as is done in [66] Section 3.1. We are interested in discretizing the continuous-time filter using a forward-Euler discretization and retrieving the discrete-time filter.

3.8.1 Forward Euler Discretization of the Continuous-Time Kalman Filter

We begin with the continuous-time Kalman–Bucy filter for the linear system

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Gw(t) \\ y(t) &= Cx(t) + v(t), \end{aligned}$$

where $w(\cdot)$ and $v(\cdot)$ are white, zero-mean noises with covariances $Q\delta(t - \tau)$ and $R\delta(t - \tau)$, respectively. The continuous-time Kalman filter equations are

$$\dot{\hat{x}}(t) = A\hat{x}(t) + K(t)[y(t) - C\hat{x}(t)] \tag{3.45}$$

$$\dot{P}(t) = AP(t) + P(t)A^\top + Q - P(t)C^\top R^{-1}CP(t) \tag{3.46}$$

$$K(t) = P(t)C^\top R^{-1}. \tag{3.47}$$

We also specify the initial conditions

$$\begin{aligned}\hat{x}(0) &= \hat{x}_0 \\ P(0) &= P_0.\end{aligned}$$

To discretize these equations at sampling instants $t_k = kh$ (with fixed step size $h > 0$), we define:

$$\hat{x}_k := \hat{x}(t_k), \quad P_k := P(t_k), \quad K_k := K(t_k) = P_k C^\top R^{-1}, \quad y_k := y(t_k).$$

A forward Euler approximation replaces each continuous derivative by a finite difference.

$$\frac{\hat{x}_{k+1} - \hat{x}_k}{h} \approx A\hat{x}_k + K_k [y_k - C\hat{x}_k].$$

Hence,

$$\hat{x}_{k+1} = \hat{x}_k + h \left[A\hat{x}_k + K_k (y_k - C\hat{x}_k) \right].$$

Similarly,

$$\frac{P_{k+1} - P_k}{h} \approx AP_k + P_k A^\top + Q - P_k C^\top R^{-1} C P_k,$$

giving

$$P_{k+1} = P_k + h \left[AP_k + P_k A^\top + Q - P_k C^\top R^{-1} C P_k \right].$$

Meanwhile, in discrete form equation (3.47) is:

$$K_k = P_k C^\top R^{-1}.$$

The initial conditions are just

$$\hat{x}_0 = \hat{x}(0), \quad P_0 = P(0).$$

Putting the equations together yields the following discrete-time forward-Euler approximation of the continuous-time Kalman filter:

$$\begin{aligned}\hat{x}_{k+1} &= \hat{x}_k + h \left[A\hat{x}_k + K_k (y_k - C\hat{x}_k) \right] \\ P_{k+1} &= P_k + h \left[AP_k + P_k A^\top + Q - P_k C^\top R^{-1} C P_k \right]\end{aligned}$$

$$K_k = P_k C^\top R^{-1}.$$

See Chapter 8 in [95] for a thorough presentation on the relation between the continuous time formulation of the Kalman filter and its discrete counterpart. See Sections 2.1 and 2.2 in [73] and Chapter 5 of this thesis for more on the consistency between the continuous and discrete time versions of the MEE.

Remark 9. Unlike the standard discrete-time Kalman filter, these equations do not separate neatly into a predict step and an update step. Instead, each forward-Euler increment effectively does a small amount of continuous “prediction” and measurement “correction” at once. For large h , numerical stability can become an issue, and more sophisticated methods such as exact matrix exponential discretization are often preferred in practice. \square

3.9 Approximate Approaches to Estimation

3.9.1 Extended Kalman Filter

We work with the discrete-time nonlinear system (3.1) and cost function (3.2) with noises $v_n \in \mathbb{R}^n$ and $w_n \in \mathbb{R}^l$ are white, zero-mean, uncorrelated noises with respective covariance matrices $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{l \times l}$.

Set $\hat{x}(n|\tau)$ to be the Extended Kalman Filter (EKF) estimate and let $P(n|\tau)$ be the approximate error covariance; τ here is either $n - 1$ or n .

Assume $\hat{x}(n|n)$ and $P(n|n)$ are known. The EKF prediction step is:

$$\hat{x}(n+1|n) = f(\hat{x}(n|n)) \tag{3.48}$$

$$P(n+1|n) = F(n)P(n|n)F^\top(n) + g(n)Q(n)g^\top(n) \tag{3.49}$$

where $F(n) = \frac{\partial f}{\partial x}(n, \hat{x}(n, n))$.

The assimilation step of the EKF is given by:

$$\hat{x}(n|n) = \hat{x}(n|n-1) + K(n)(y(n) - h^{[0]}(n)) \tag{3.50}$$

$$P(n|n) = (I - K(n)H(n))P(n|n-1) \tag{3.51}$$

where

$$h^{[0]}(n) = h(n, \hat{x}(n|n-1))$$

$$H(n) = \frac{\partial h}{\partial x}(n, \hat{x}(n|n-1)).$$

And the Kalman gain $K(n)$ is:

$$K(n) = P(n|n-1)H^\top \left(R + H(n)P(n|n-1)H^\top(n) \right)^{-1}.$$

When the dynamics are highly nonlinear, the accuracy of the extended Kalman filter greatly diminishes.

3.9.2 Krener's Taylor Polynomial discrete-time MEE scheme

We now introduce Krener's work on discrete-time MEE [59]. A dynamic programming approach is taken but Taylor series approximations are then introduced in order to simplify the necessary calculations. It is also shown in the paper that if the series expansion degree $d = 1$, then the method reduces to the extended Kalman filter but when the higher degree approximations are taken, the estimates improve.

Again we are working with the same system (3.1) and cost function (3.2) and we again set $x(n) = \xi$ for a given n . This is a fixed point.

The estimation procedure is as follows:

Step 1: We assume that $V(\xi, 0|0)$ and $\hat{x}(0|0) = \operatorname{argmin}_\xi V(\xi, 0|0)$ are known. In a computational setting the initial estimation can be guessed.

Step 2: Given $V(\xi, n|n)$ we can compute $V(\xi, n+1|n)$ by solving:

$$V(\xi, n+1|n) = \min_{x,v} \left(V(\xi, n|n) + \frac{1}{2} \|v\|^2 \right) \quad (3.52)$$

subject to:

$$x_{s+1} = f x_s + g v_s. \quad (3.53)$$

Step 3: Having computed $V(\xi, t+1|t)$ we can solve

$$\hat{x}(t+1|t) = \operatorname{argmin}_\xi V(\xi, t+1|t) \quad (3.54)$$

to obtain the estimate $\hat{x}(t+1|t)$.

Step 4: We solve

$$V(\xi, t+1|t+1) = V(\xi, t+1|t) + \frac{1}{2} \|y(t+1) - h(t+1, \xi)\|_{R^{-1}}^2 \quad (3.55)$$

to finally obtain:

$$\hat{x}(t+1|t+1) = \underset{\xi}{\operatorname{argmin}} V(\xi, t+1|t+1). \quad (3.56)$$

which completes Step 5.

We then increment t by one and restart at step 2. Although this algorithm is clear, in practice it is difficult to implement. A Taylor polynomial approach is taken in order to solve it. For the full details, see [59].

But the steps are simplified as follow.

Step 1: remains the same

Step 2:

$$V^{[0:d+1]}(\xi, t+1|t) = V^{[0:d+1]}(\bar{x}^{[0:d]}(x), t|t) + \frac{1}{2} \|v^{[1:d]}(\xi)\|^2.$$

where $V^{[0:d+1]}(\xi, t|t)$ is the Taylor approximation of $V(\xi, t|t)$ of degree $d+1$ around $\bar{x} = \xi - \hat{x}(t+1|t)$.

Step 3: can be computed using $\hat{x}(t|t)$ and f

$$\hat{x}(t+1|t) = f(\hat{x}(t|t)).$$

Step 4:

$$V^{[0:d+1]}(\xi, t+1|t+1) = V^{[0:d+1]}(\xi, t+1|t) + \frac{1}{2} (\|y(t+1) - h^{[0:d]}(t+1, \xi)\|_{R^{-1}}^2)^{[0:d+1]}.$$

Step 5:

$$\hat{x}(t+1|t+1) = \underset{\xi}{\operatorname{argmin}} V^{[0:d+1]}(\xi, t+1|t+1).$$

We then increment t by one and restart at step 2.

When taking $d = 1$, meaning we linearize the dynamics, this approach reduces to the extended Kalman filter, as shown in [59].

3.9.3 Unscented Kalman Filter

In cases where $f(\cdot)$ and $h(\cdot)$ are highly nonlinear, the reliance of the extended Kalman filter on first-order linearizations can lead to inaccuracies. The unscented Kalman filter provides an alternative that avoids explicit linearization by leveraging the unscented transform.

The unscented Kalman filter operates under the premise that it is easier to approximate a Gaussian distribution than to approximate nonlinear functions. To this end, the unscented Kalman filter uses a deterministic sampling approach to propagate the mean and covariance through the nonlinear system dynamics and measurement models. These samples, called sigma points, are chosen to capture the mean and covariance of the prior distribution exactly and are then propagated through the nonlinear functions.

The unscented Kalman filter typically outperforms the extended Kalman filter in scenarios with significant nonlinearity due to its higher-order approximation of the state distribution. However, it incurs a greater computational cost due to the generation and propagation of sigma points.

See page 449 in [95] for more details on the unscented Kalman filter.

3.9.4 Particle Filter

In cases where $f(\cdot)$ and $h(\cdot)$ are highly nonlinear or when the noise distributions deviate significantly from a Gaussian distribution, both the extended Kalman filter and the unscented Kalman filter can suffer from performance issues. Particle filtering provides a general framework for handling such cases by using a Monte Carlo approach to approximate the posterior distribution of the state.

Particle filters operate under the premise that any probability distribution can be approximated by a set of discrete samples, or particles, and associated weights. These particles are propagated through the nonlinear system dynamics, and their weights are updated based on the likelihood of the observations. Together, the particles and weights provide an empirical approximation of the posterior distribution of the state.

Particle filters are highly flexible and can handle a wide range of nonlinear and non-Gaussian systems. However, their accuracy depends on the number of particles N , which directly impacts computational cost. For high-dimensional systems, the particle filter may suffer from the curse of dimensionality, requiring a prohibitively large number of particles to maintain accuracy.

Compared to Kalman-based approaches, particle filters provide a more accurate approximation of the state distribution in complex scenarios but at the cost of increased computational complexity.

See page 468 in [95] for more details on the particle filter.

Chapter 4

Numerical Methods for Solutions to the HJB

Solving HJB equations is crucial to solving problems in optimal control and optimal estimation. The HJB equation can be circumvented by finding open-loop controls; however in most applications these are not acceptable controls due to their lack of stability when there are external disturbances and thus most problems require closed loop controls; getting solutions to the HJB is thus crucial. For a review of open-loop methods the reader can check the introduction in [11]. In this section we want to give an overview of the many numerical methods and techniques that exist to compute solutions to the HJB. Some methods use classical algorithms, others use deep learning or neural networks, some a combination of many techniques. An exhaustive review would take more than a chapter but hopefully this chapter will give the reader an overview of this very active area of research.

The first and main difficulty in solving the HJB numerically is known as the “curse of dimensionality”. Coined by Bellman in [12], this expression refers to the fact that the computational cost of the commonly employed grid-based numerical used to solve these equations, grows exponentially with the order of the system. Examples of such methods include finite elements methods [22], [64], [48], finite differences schemes [65] and level set methods [83], [92], [82]. The curse of dimensionality renders their practical implementation limited to low order systems. If a naive approach is taken, the computations for a system with dimension d greater than $d = 3$ [103] will be unfeasible, even with a supercomputer.

4.1 Classical Methods

Some grid-based methods exploit aspects of the problem in order to circumvent the curse of dimensionality. For example Bokanowski et al. [18] developed a spatially adaptive semi-Lagrangian sparse grid scheme for solving time-dependent HJB equations. Semi-Lagrangian schemes take advantage of the best of both Eulerian and Lagrangian frameworks. Eulerian schemes work well on regular meshes but lead to restrictive time steps for stability, Lagrangian schemes use much larger time steps but differences in initial conditions lead to large differences later in the simulation, meaning, for example, that a small error in the guess of the initial condition can lead to large discrepancies later in the simulation. The semi-Lagrangian scheme takes the best of both worlds: “the regular resolution of Euler schemes and the enhanced stability of Lagrangian ones” [97]. In [18], the method handles high-dimensional problems (up to 8 dimensions) efficiently by leveraging adaptive sparse grids and specialized boundary treatments to reduce grid points. The authors note “the proposed scheme neither has the monotony property that would give convergence towards the viscosity solution, nor has provable stability as far as we know. However, we think that this initial work is an encouraging step towards the construction of related schemes for the solution of HJB equations in higher dimensions that could remedy these drawbacks.” In [38], Falcone and Ferretti take an extensive look at semi-Lagrangian methods for PDEs, with chapter 8 focusing on applications in control.

The authors remark that despite the increasing number of accurate and efficient high-order schemes, the convergence theory accompanying them is lacking, Chapter 6 of their book provides a theoretical analysis of high-order schemes for HJBs.

Cacace, Cristiani, Falcone and Picarelli in [20], presented a patchy dynamic programming scheme for efficiently solving HJB equations arising in optimal control problems. Ancona and Bressan first introduced the notion of “patchy” methods in [6] in a purely theoretical setting. The main result states that under suitable assumptions, a control system can be stabilized via a piecewise constant patchy feedback on a particular partition of the domain. The important property of each part is that it is invariant with respect to the optimal dynamics, meaning that once a patchy decomposition is obtained, the computations on each can be parallelized. The application of the method in [20] combines classical dynamic programming principles with the patchy method. By applying high-order interpolation techniques within each patch and coupling the solutions across patches, the scheme is both efficient and accurate. Krener and Navasca have developed a different patchy method for discrete-time systems as well [79], [80], [46].

Often, model order reduction techniques [5] are used; they reduce the complexity of the system and, therefore, the dimension of the corresponding HJB equation. The reduced-

basis method [85] and proper orthogonal decomposition [15] and [61] are two actively used techniques that achieve model reduction. In these methods, the order reduction of the size of the problems to be solved is typically achieved by using pre-computed solutions to generate a well-chosen subspace. Within this subspace, the solution is then approximated [23]. In [23], the method is used to compute an HJB based feedback control, in [62], it is applied to observers.

Methods that entirely avoid grids have also been developed in order to avoid the curse of dimensionality entirely. Generalized Hopf–Lax formulas can be used to solve the HJB equations in a tractable grid free manner without the need for numerically optimizing the Hamiltonian [21]. These methods rely on the generalized Hopf–Lax formulas to transform the computation of the value function at an arbitrarily given space-time point into an optimization problem for the terminal value of the Lagrangian multiplier p subject to the characteristics equation of Hamiltonian ordinary differential equation for (x, p) as reviewed in [14]. The method of characteristics and Hopf formula-based methods were initially used for state-independent HJB equations but they were later extended, under certain conditions for state-independent HJBs [104]. Darbon and Osher, in [29], propose and test methods for solving HJBs without the use of grids or numerical approximations. Instead, they use the classical Hopf formulas [44] for solving initial value problems resulting in methods that “are polynomial in the dimension. [They] carefully explain how to compute numerically the optimal control from the numerical solution of the associated initial valued HJ PDE for a class of optimal control problems” [29]. In 2019, the same authors along with Chow and Yin extend their results in [25]. They “conjecture a (Lax-type) minimization principle to solve state-dependent HJ PDE when the Hamiltonian is convex, as well as a (Hopf-type) maximization principle to solve state-dependent HJ PDE when the Hamiltonian is non-convex, as a generalization of the well-known Hopf formula”. Yegorov in [103], [104] further extends the results and by using the method of characteristics, evaluates problems with different convexity/concavity conditions on the problem’s Hamiltonian.

In [3] and [91], Alla, Saluzzi and Falcone introduce a tree structure approach to solving finite horizon optimal control problems. The discrete time dynamics construct a tree structure algorithm. Although this method eliminates the space discretization and grid construction, which allows for computation of feedback controls in dimension $d = 1000$, the tree structure requires many PDEs to be solved for each control input. In [4], Alla and Saluzzi, combine the tree structure algorithm with proper orthogonal decomposition [102] in order to speed up the method.

Methods based on tensor decomposition [17], [56], [41] techniques have also been popular. Tensor-train formats are low-rank structures which are used to represent high-dimensional objects, for a first introduction check [81]. Dolgov, Kalise and Kunisch in [30] use this

technique to avoid the curse of dimensionality resulting in computations for dimensions $d > 100$, the computational costs have polynomial scaling with respect to the dimension. Tensor-train methods have also been used in [99], [45], [90] and [89].

Another class of popular methods include max-plus (or min-plus) [2] schemes. In this method, max-plus algebras are used to represent the dynamic systems of interest in a simplified manner, this alleviates computational burden. The following papers apply this technique to optimal control: [70], [33]. It is applied to nonlinear filtering in [39].

4.2 Neural Network Methods

Much work has been done in using neural networks and deep learning methods for solving PDEs and more specifically solving the the HJB [10], [8]. An exhaustive review would be too long for this thesis and is out of scope. We will limit our review to papers using neural networks or deep learning and specifically applying the techniques to find optimal controls, not simply solving the HJB; however, the reader is encouraged to check [34] for a great review of the different methods that exist.

Recht [87], Vamvoudakis, Lewis and Ge [101] and Bertsekas [16] have great overviews of how reinforcement learning and neural networks are used in the context of optimal control. Often, neural network based techniques minimize the residual of the HJB and have artificial boundary conditions at randomly sampled collocation points [1], [24], [98] and [96]. In [49] and [47], the control and the HJB solution and its gradient are computed around a trajectory. The successive approximation method reduces the non linear HJB into a sequence of linear PDEs called the Generalized Hamilton-Jacobi-Bellman (GHJB) equation. Often, the Galerkin method is used to solve these problems for low dimensions. As seen before, the curse of dimensionality can be avoided for moderately large dimension problems, for example Kalise and Kunisch use polynomial approximations [50] for problems with dimension up to $d = 14$ but for higher dimension problems, neural networks are required [14].

Another class of methods are data-driven. Data sets created using causality-free methods such as algorithms based on the link between PMP or a State-Dependent Riccati equation strategy, which employs generalized solutions to the Riccati equation for nonlinear dynamics [31]. Causality-free methods are usually too slow for online computation, but they are perfectly parallelizable so can be used to generate large data sets offline, smaller data sets can also be computed and then with a time-marching trick, additional data can be added during the training phase [78]. In [78], for example, small set of open-loop optimal control solutions are generated using a causality-free algorithm based on PMP, and additional data

samples are added in regions where the value function is difficult to learn.

There has also been a lot of success in using Physics Informed Neural Networks (PINNs) in optimal control recently. Whereas traditional neural networks are trained on data and a loss function which penalizes the prediction error solely based on the available data, PINNs incorporate a loss function that enforces the governing equations and boundary conditions. PINNs can solve forward problems in the absence of any data when the governing equations are fully known, or they can leverage available data to solve inverse problems involving unknown model parameters or physical quantities [76]. In [72], the authors show how effective PINNs are for solving PDE's and in [68], PINNs are combined with proximal-policy optimization to solve the HJB and in [77], Nakamura-Zimmerer, Gong and Kang, include an LQR approximation for the linearized dynamics in the neural network value function. In [28], Darbon, Dower and Meng combine min-plus algebra (the set of real numbers endowed with plus and min operators) [86] and neural networks to solve high-dimensional optimal control problems. Deep neural networks are used in [63] to find semi global optimal controls. Using a similar method, Breiten and Kunisch in [19] focus on using neural networks for nonlinear observers. The authors avoid computing the observer gain directly since it involves taking the inverse of the Hessian of the solution to the HJB equation. This is not feasible in practice and thus they take an alternate approach which doesn't depend on the solution of the HJB. They use a neural network to directly compute the gradient of the HJB, i.e the value function. Their method directly approximates observer gains by replacing the disturbance v in the dynamics by a neural network. They state that in future research, they will provide detailed analysis and rigorous proofs, as well as advance code development for treating higher dimensional problems.

Chapter 5

Simulations

We now present simulations conducted using Matlab. First there is a comparison of a discrete time Kalman filter and a “hybrid” Kalman filter - meaning a continuous time filter with a late discretization - applied to a mass-spring system. Then, the accuracy of the discrete time Kalman filter applied to this system is tested using different factors in front of the disturbance covariance R . Finally the comparison of a discrete time Kalman filter and a “hybrid” Kalman filter is applied to a system of five masses and springs.

With these simulations we hope to investigate whether an early or late discretization of the filter has any impacts on estimation accuracy.

5.1 Discrete Time vs “Hybrid” Kalman Filter for Mass Spring System

We first set up the linear system and solve it for $t \in [0, 10]$ using Matlab’s built-in ODE solver “ode45” based on an explicit Runge-Kutta (4,5) formula [32], [93]. We take periodic measurements and add random disturbances to them. Then, we apply the discrete time Kalman filter estimating the position and velocity of the simulated mass; this estimation is done every time a new measurement becomes available. We also apply the continuous time Kalman filter, leading to a Riccati equation which is solved by discretization, this is dubbed the “hybrid” filter.

Let’s provide a closer look at the MATLAB script used to compare the discrete-time and “hybrid” Kalman filters on a single mass-spring system. While the full code appears in

Appendix B, here we highlight its most important steps and discuss how each approach is implemented.

We first define parameters for a simple mass-spring-damper system by setting $m = 1$, $b = 0.3$, $k = 10$, and $u = 0.5$. These values respectively represent the mass, damping, spring constant, and the constant force applied to the mass. We then write the dynamics in state-space form as

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu \\ y(t) &= Cx(t) + v(t),\end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix}, \quad B = \begin{bmatrix} b/m \\ 0 \end{bmatrix}, \quad C = [1 \quad 0],$$

$x(t) = [\text{position, velocity}]^\top$ and $v(t)$ is zero-mean random disturbances. We use `ode45` to solve these equations for $t \in [0, 10]$, creating a “true” trajectory for position and velocity against which we can compare filter estimates.

We then introduce a measurement interval $\Delta t = 0.4$ and sample the true state at times $t_{\text{meas}} = 0, 0.4, 0.8, \dots, 10$. To simulate noisy data, we add Gaussian noise to the measured position, obtaining

$$y_{\text{meas}}(t_{\text{meas}}) = x_{\text{true}}(t_{\text{meas}})(\text{position}) + r\varepsilon,$$

with $\varepsilon \sim \mathcal{N}(0, 1)$ and r the noise standard deviation. The code also defines process disturbance covariance Q and measurement disturbance covariance R , set to r^2 , with an additional factor of 5 to stress-test the filter. Later simulations test the difference in estimation accuracy with different factors in front of the disturbance covariance R .

After this set up, we implement two Kalman Filters. The discrete-time Kalman filter relies on discretizing the continuous model from t_k to t_{k+1} . We compute the discretized matrices for the dynamics and the covariances as in [95]:

$$\begin{aligned}A_d &= e^{A\Delta t}, \\ B_d &= A^{-1}(A_d - I)B,\end{aligned}$$

and scale Q and R by Δt so $Q_d = Q\Delta t$ and $R_d = R\Delta t$.

At each step, we perform a prediction update:

$$\begin{aligned}x_k^- &= A_d x_{k-1}^+ + B_d u, \\ P_k^- &= A_d P_{k-1}^+ A_d^\top + Q_d,\end{aligned}$$

followed by the usual measurement update:

$$\begin{aligned}
K_k &= \frac{P_k^- C^\top}{C P_k^- C^\top + R_d} \\
x_k^+ &= x_k^- + K_k (y_k - C x_k^-) \\
P_k^+ &= (I - K_k C) P_k^- (I - K_k C)^\top + K R_d K^\top.
\end{aligned}$$

This process is repeated at every measurement instant.

By contrast, the “hybrid” Kalman filter evolves the system continuously and only discretizes at measurement times. In each interval $[t_{k-1}, t_k]$, we integrate the state $\dot{x} = Ax + Bu$ and the Riccati equation $\dot{P} = AP + PA^\top + Q$ using `ode45`. At t_k , we apply the same discrete update formula,

$$\begin{aligned}
K_k &= \frac{P_k^- C^\top}{C P_k^- C^\top + R} \\
x_k^+ &= x_k^- + K_k (y_k - C x_k^-),
\end{aligned}$$

where P_k^- is taken from integrating \dot{P} . This scheme keeps the dynamics in continuous form until a new measurement arrives, an approach we’ll refer to as “late discretization”. Note the Kalman gain update in continuous time is $K = PC^\top R^{-1}$. In our simulations, if this equation was used, the estimation would blow up; our hypothesis is that this is due to the fact that the equations have already been discretized by that step.

Finally, the script computes the estimated state trajectories from each method and compares them to the true solution, both visually and by calculating the root mean squared error for position and velocity. The “hybrid” filter often shows slightly improved accuracy because it more faithfully follows the continuous dynamics, whereas the discrete-time approach uses matrix exponentials and updates the state and covariance in jumps. As the measurement interval Δt grows, these differences can become more pronounced, highlighting the potential advantage of solving the Riccati equation continuously.

The root mean squared error in the position for the discrete Kalman filter is 0.1162 while for the “hybrid” Kalman filter it is 0.1052. This difference of around 10% is consistent across dozens of simulations.

Although we don’t have direct measurements for the velocity, the estimates are still accurate.

The root mean squared error in the velocity for the discrete Kalman filter is 0.2373 while for the “hybrid” Kalman filter it is 0.2013. This difference of around 15% is consistent across dozens of simulations.

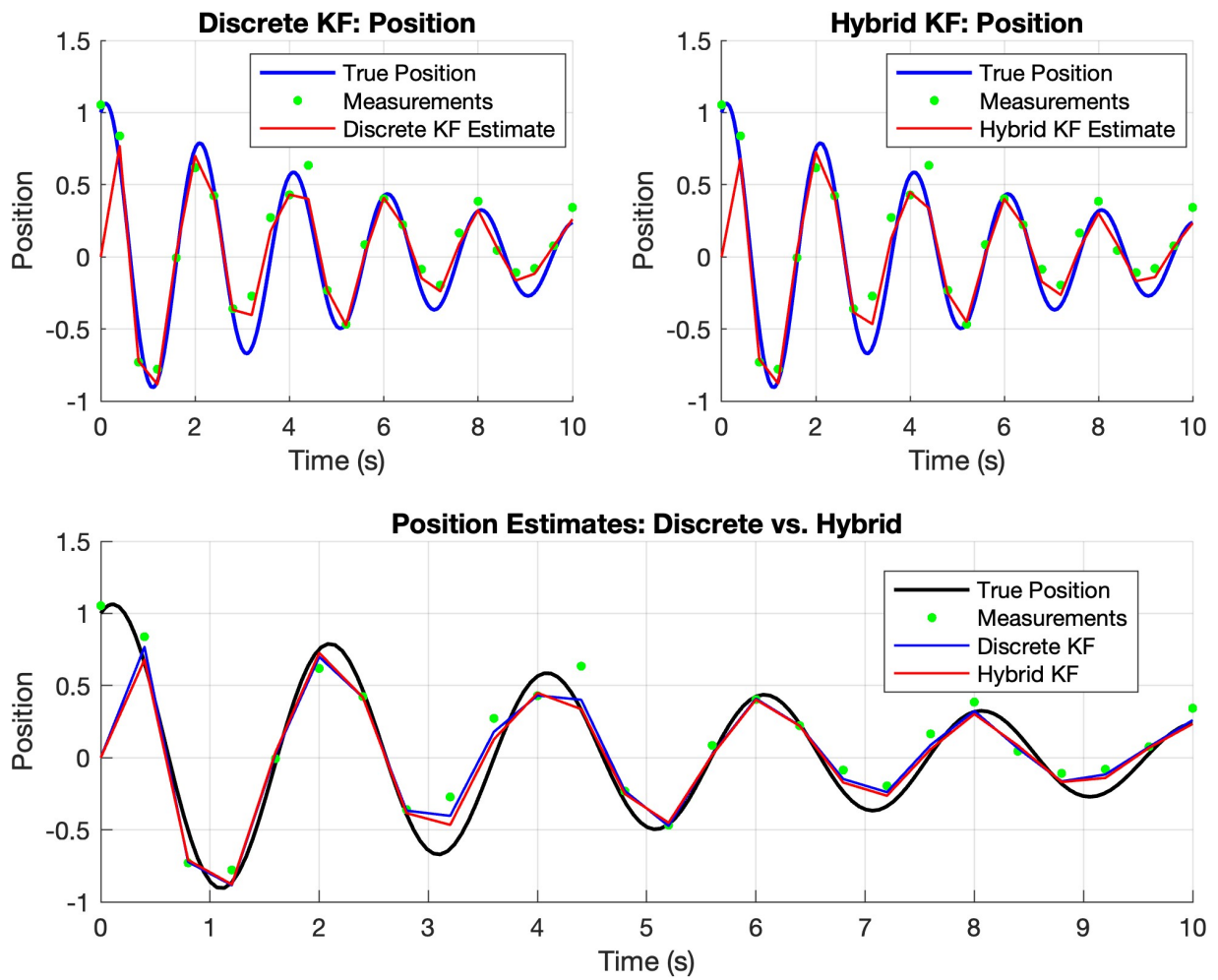


Figure 5.1: Mass Spring Position Estimates Comparison.

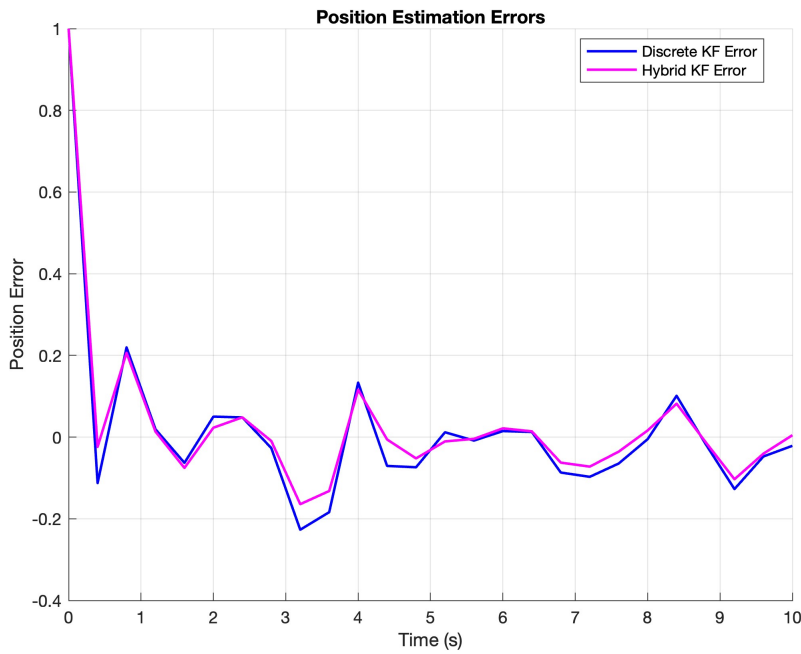


Figure 5.2: Mass Spring Position Estimates Errors.

The late discretization yields slightly better estimates. This could be due to the fact that the “hybrid” filter follows the continuous dynamics while the discrete filter follows a discretization of the dynamics. The way that the covariance P is updated, could also be a factor. While in the “hybrid” filter, it is solved by integration every step, the discrete Kalman filter uses a recursive formula, this could cause errors to accumulate.

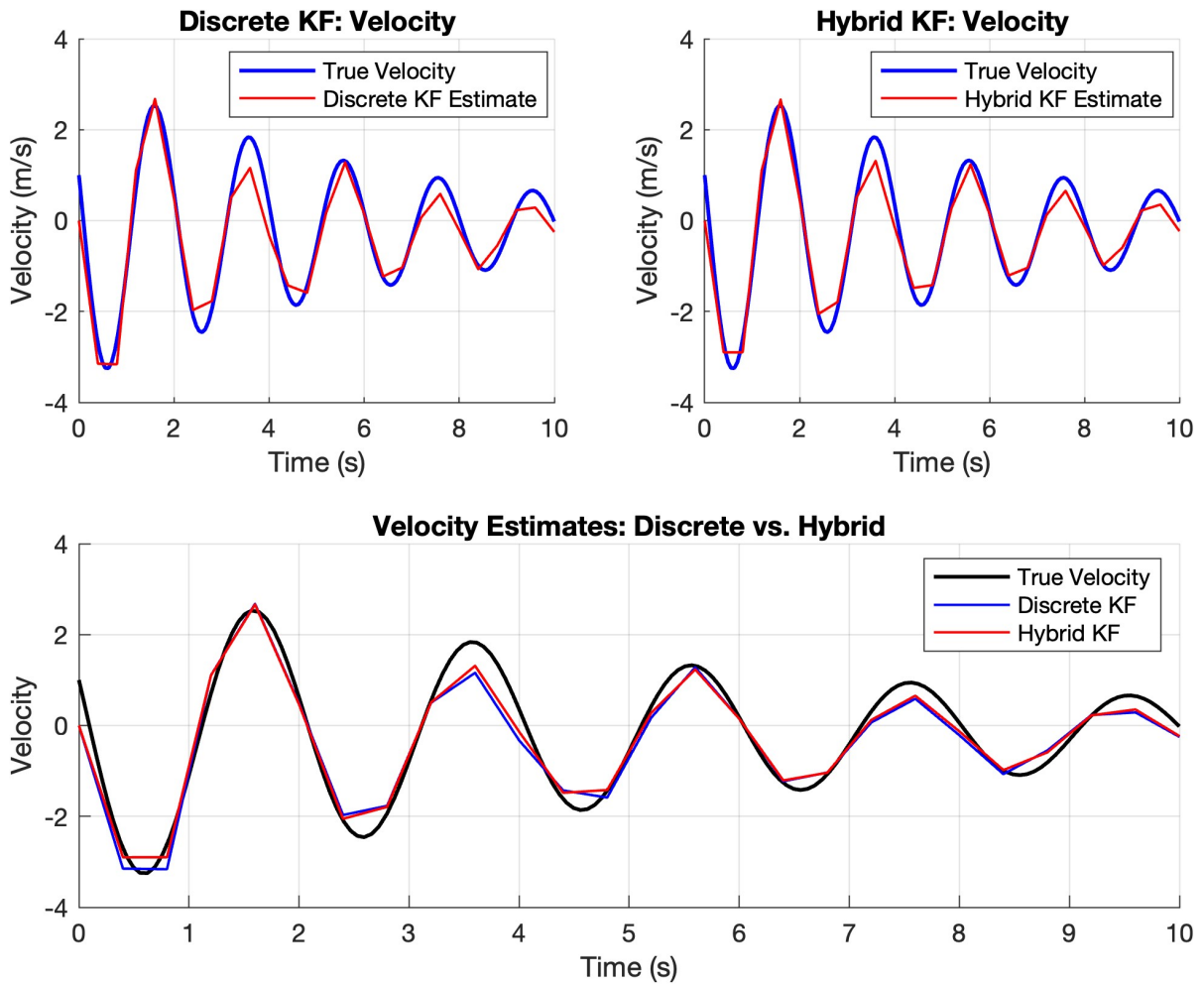


Figure 5.3: Mass Spring Velocity Estimates Comparison.

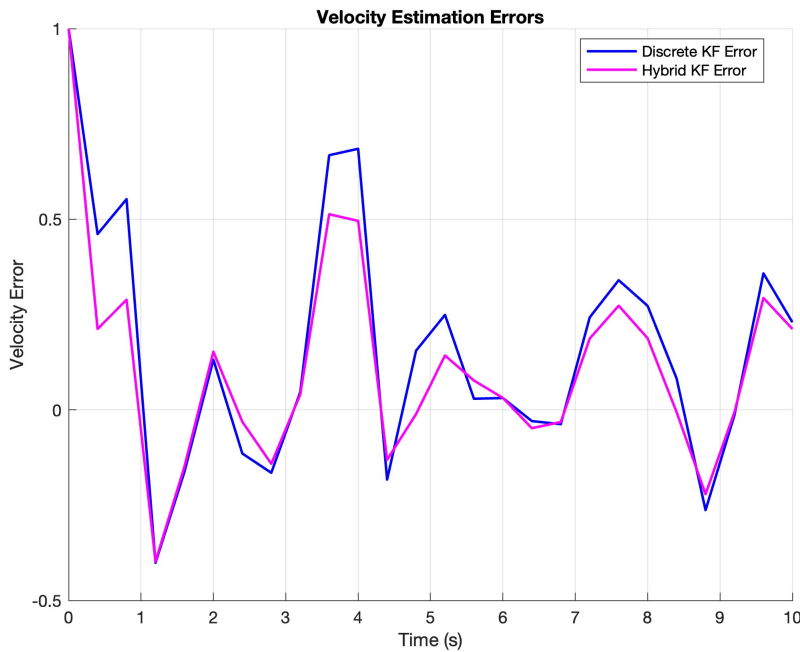


Figure 5.4: Mass Spring Velocity Estimates Errors.

5.2 Discrete Time vs “Hybrid” Kalman Filter for Multi Mass Spring System

For the higher order case we used a 0.2 time step. The code can be found in Appendix ???. We simulated a system with 5 masses connected one-by-one with a spring. All the masses and spring constants are of the same value. We only estimate the position and velocity of one mass. In this example, the difference in accuracy between the discrete and “hybrid” filter for the position of one mass is not as clear. For the velocity; however, the difference remains the same as for the mass spring case.

The root mean squared error in the position for the discrete Kalman filter is 0.1310 while for the “hybrid” Kalman filter it is 0.1272. This difference of around 2% is consistent across dozens of simulations.

The root mean squared error in the velocity for the discrete Kalman filter is 0.6421 while for the “hybrid” Kalman filter it is 0.5435. This difference of around 15% is consistent across dozens of simulations.

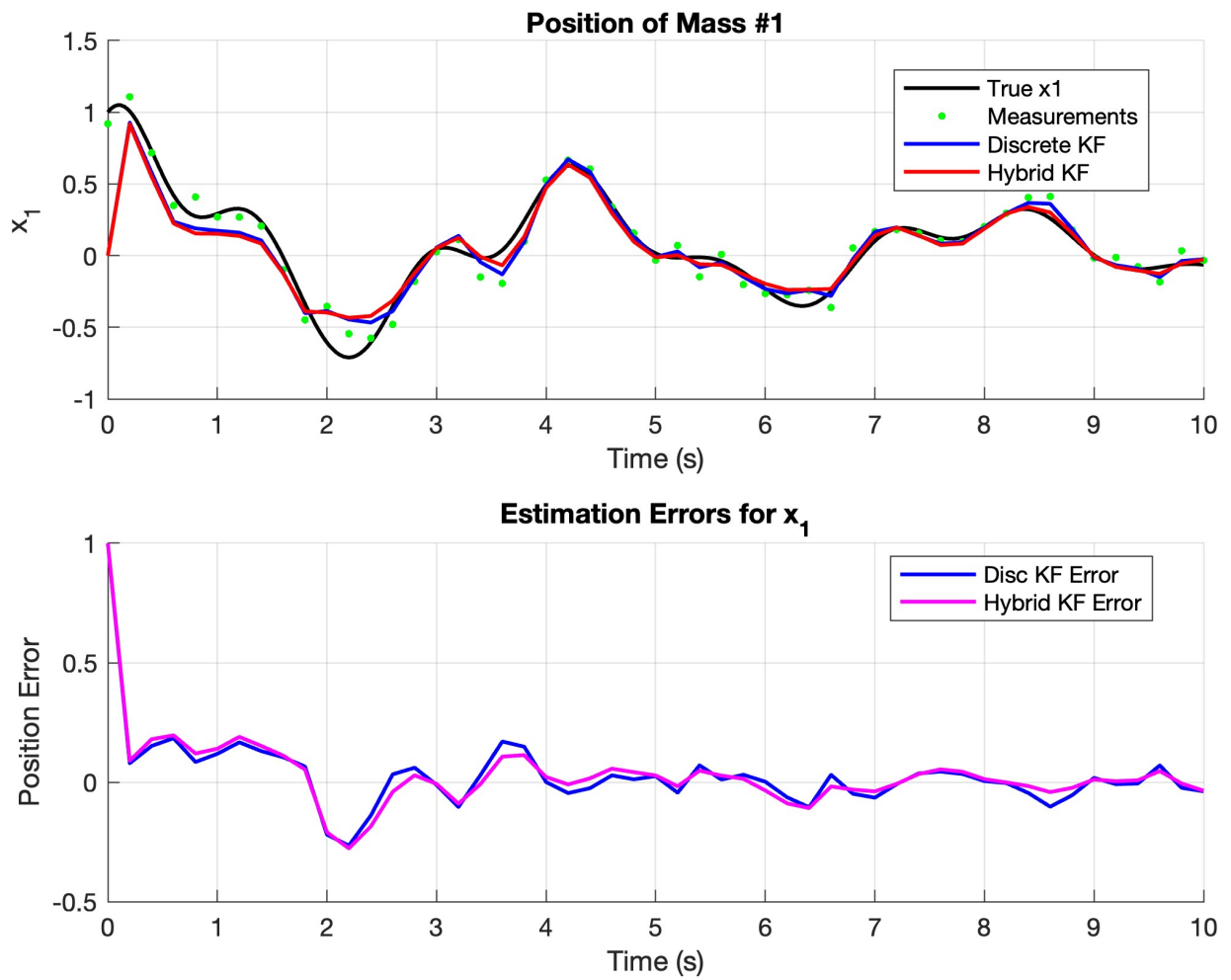


Figure 5.5: Multi Mass Spring Position Estimates and Errors.

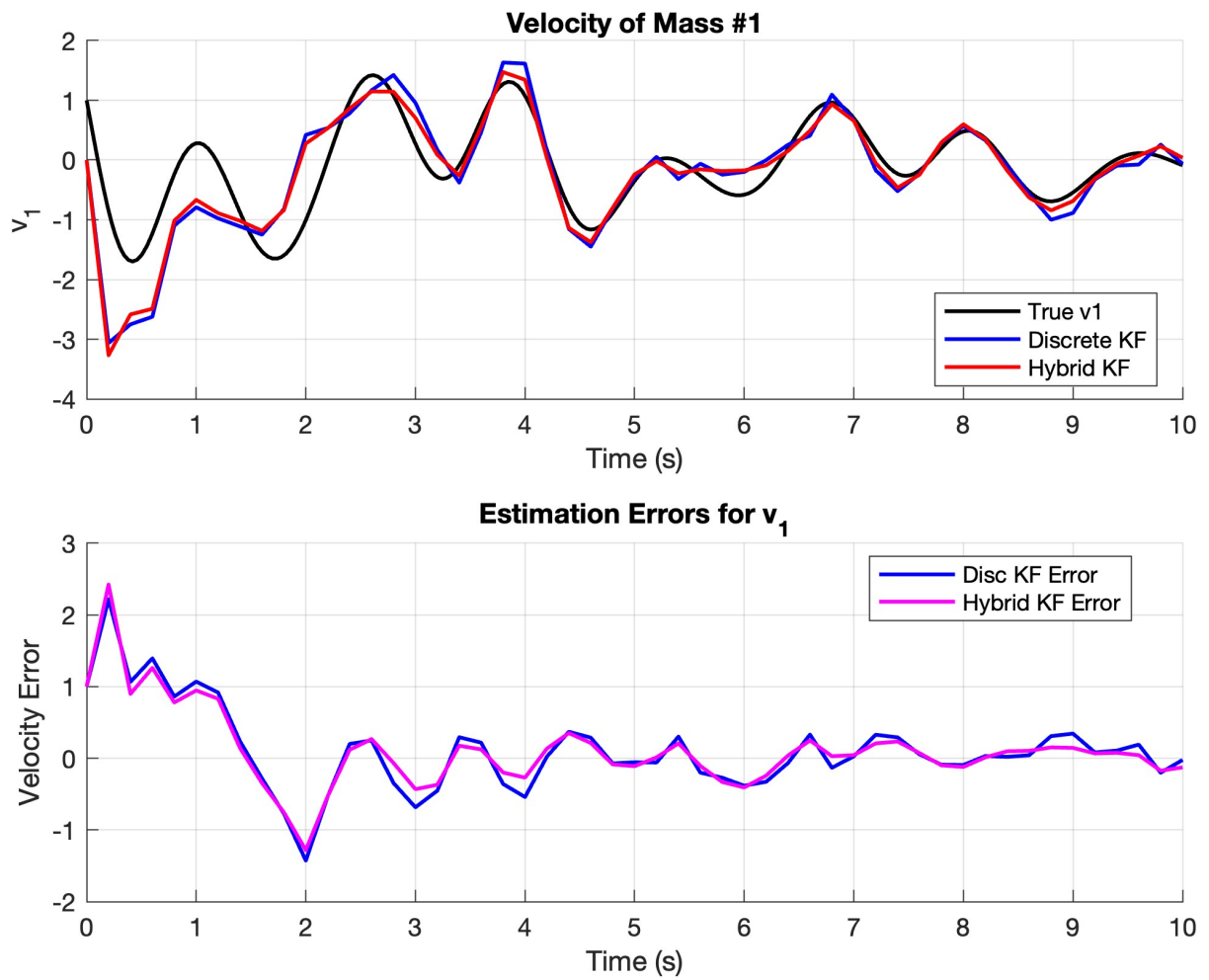


Figure 5.6: Multi Mass Spring Velocity Estimates and Errors.

5.3 Discussion on Small or Large Noise Covariance R

While conducting the simulations, it was observed that the estimates were sensitive to the noise covariance R . In our simulations, as seen in Appendix B, R was multiplied by a factor of 5 to make the estimates smoother. In this section we show how the discrete Kalman filter estimates for a mass spring system differ when the noise covariance R is multiplied by different factors.

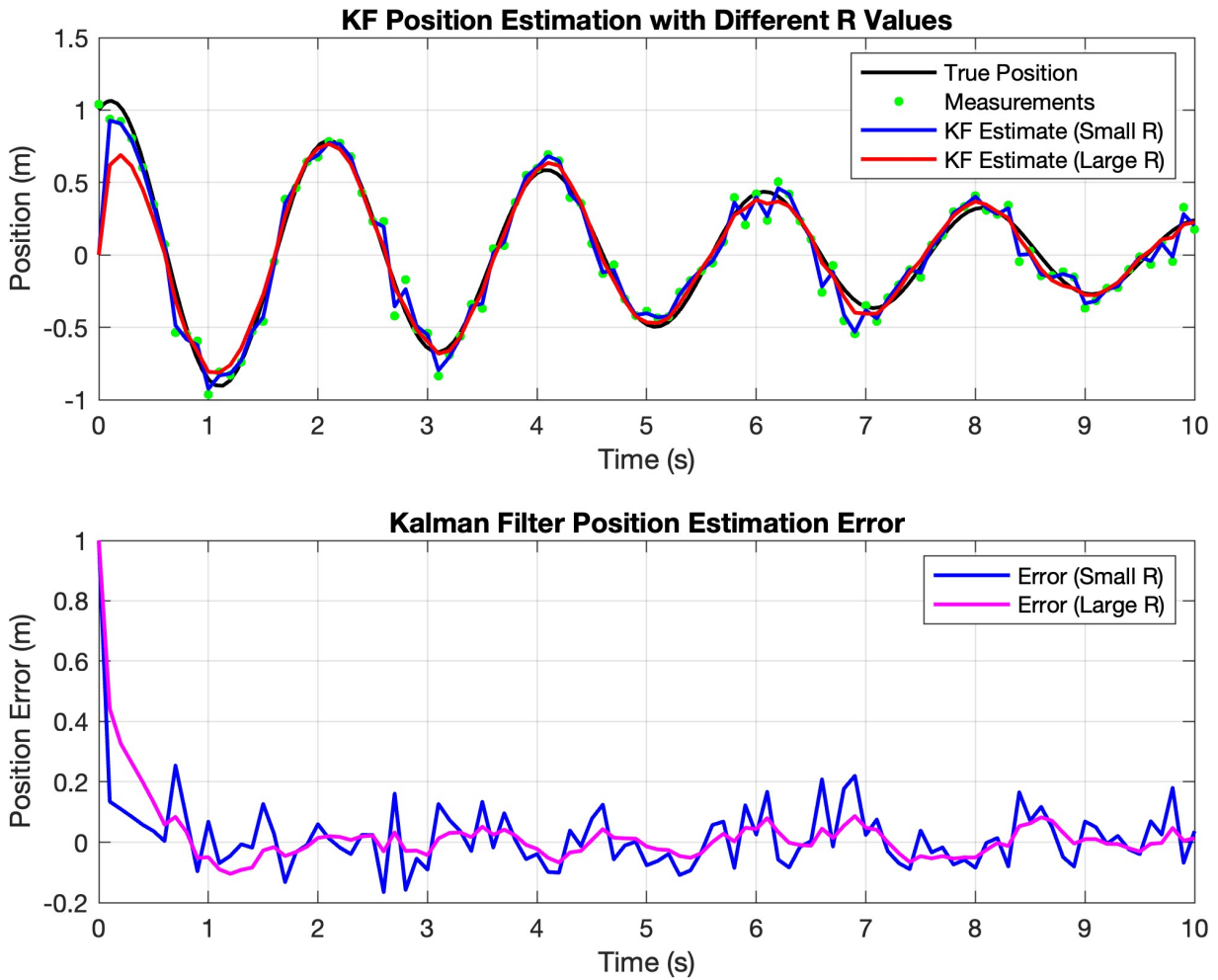


Figure 5.7: Position Estimate with Small (1) and Large (50) Factor for Noise Covariance.

In the presence of position measurements, a covariance R that is too small cause the Kalman filter to follow those measurements too closely while ignoring the model dynamics, resulting

in a jagged estimate. On the other hand, when the large factor is applied to R , the estimate is smooth and follows the dynamics closely.

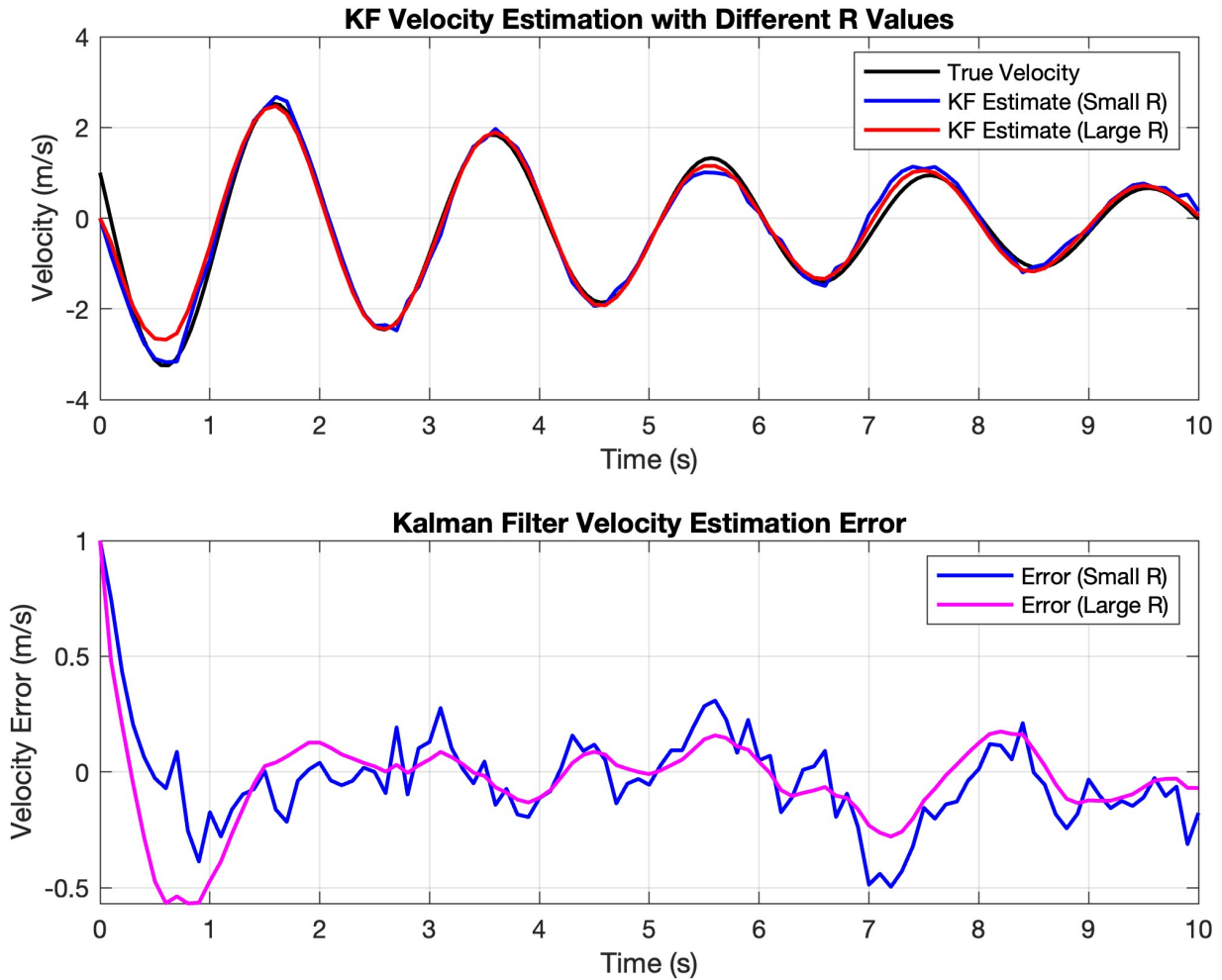


Figure 5.8: Velocity Estimate with Small (1) and Large (50) Factor for Noise Covariance.

The jagged effect is not as pronounced for the velocity since no measurements are available.

In order to investigate the issue of how covariance tuning can impact the accuracy of estimation, we also conducted the mass-spring discrete time simulations for covariance R with factors 1 through 50.

These are the resulting plots:

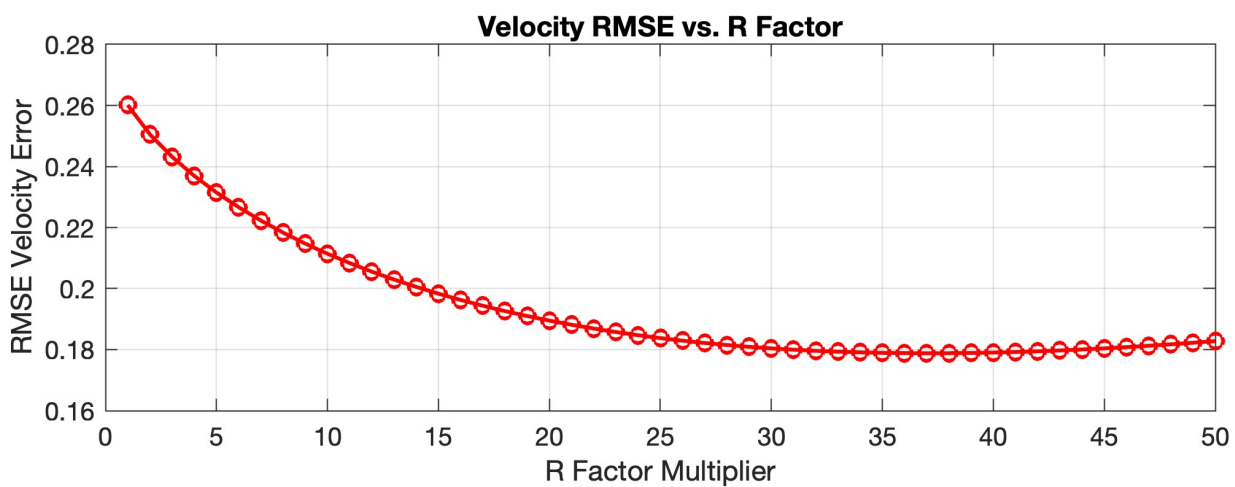
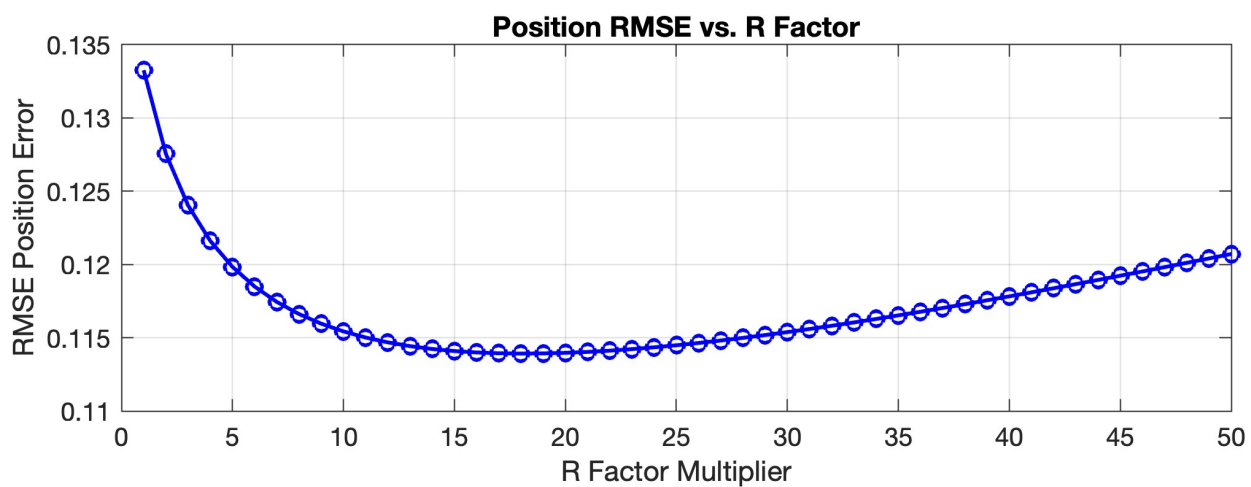


Figure 5.9: Root Mean Squared Errors for Factors 1 Through 50 for Position and Velocity Estimates.

Interestingly for the position estimates, which do take into account measurement data, the “optimal” factor for R is near 15. Increasing it further, makes the estimate less accurate.

For the velocity estimate, a similar phenomenon occurs but much less pronounced. The “optimal” factor is near 40.

The estimates for covariance factors 1, 25, and 50 are plotted below:

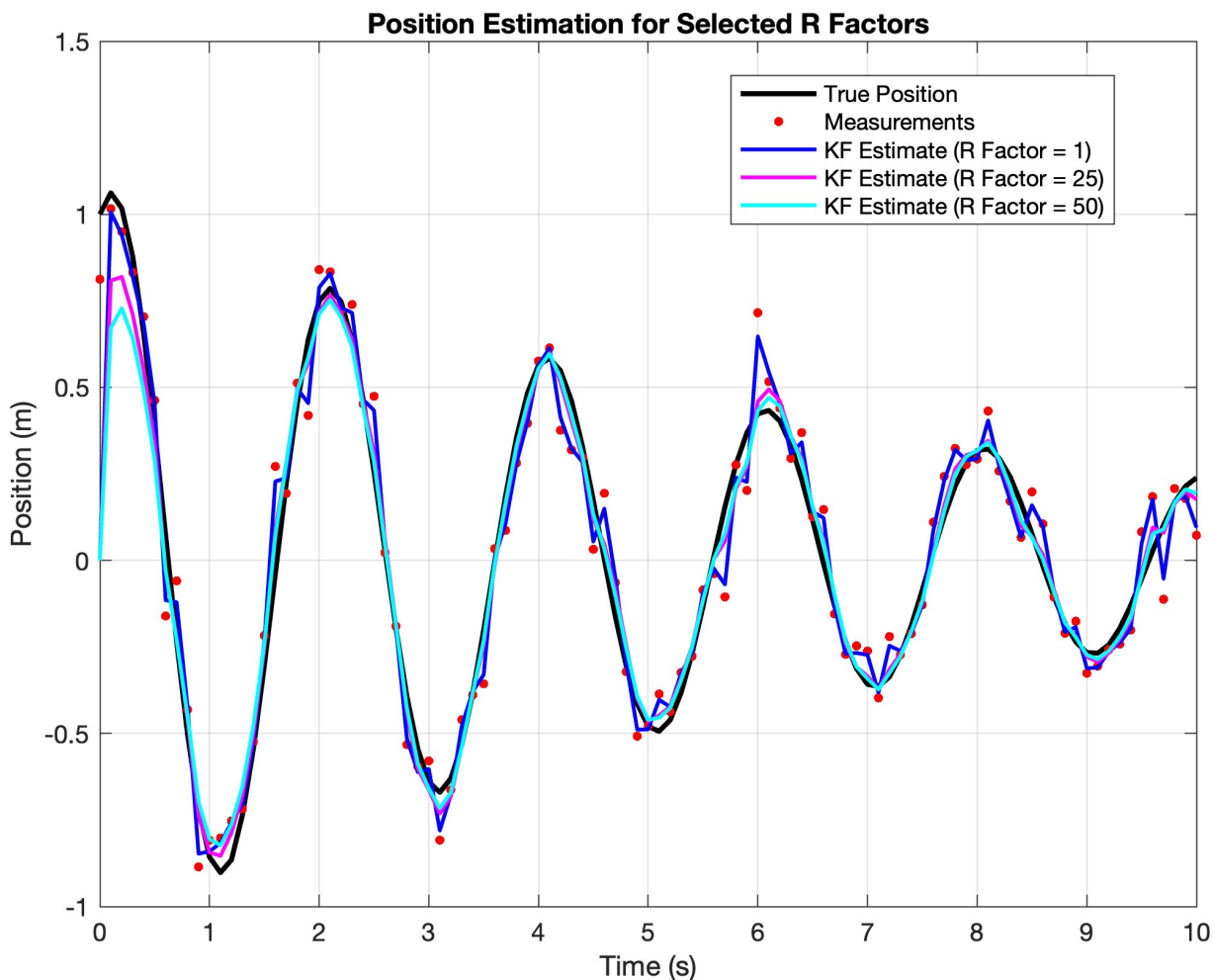


Figure 5.10: Position Estimates for Mass Spring System for Three Different Factors Multiplying the Covariance R .

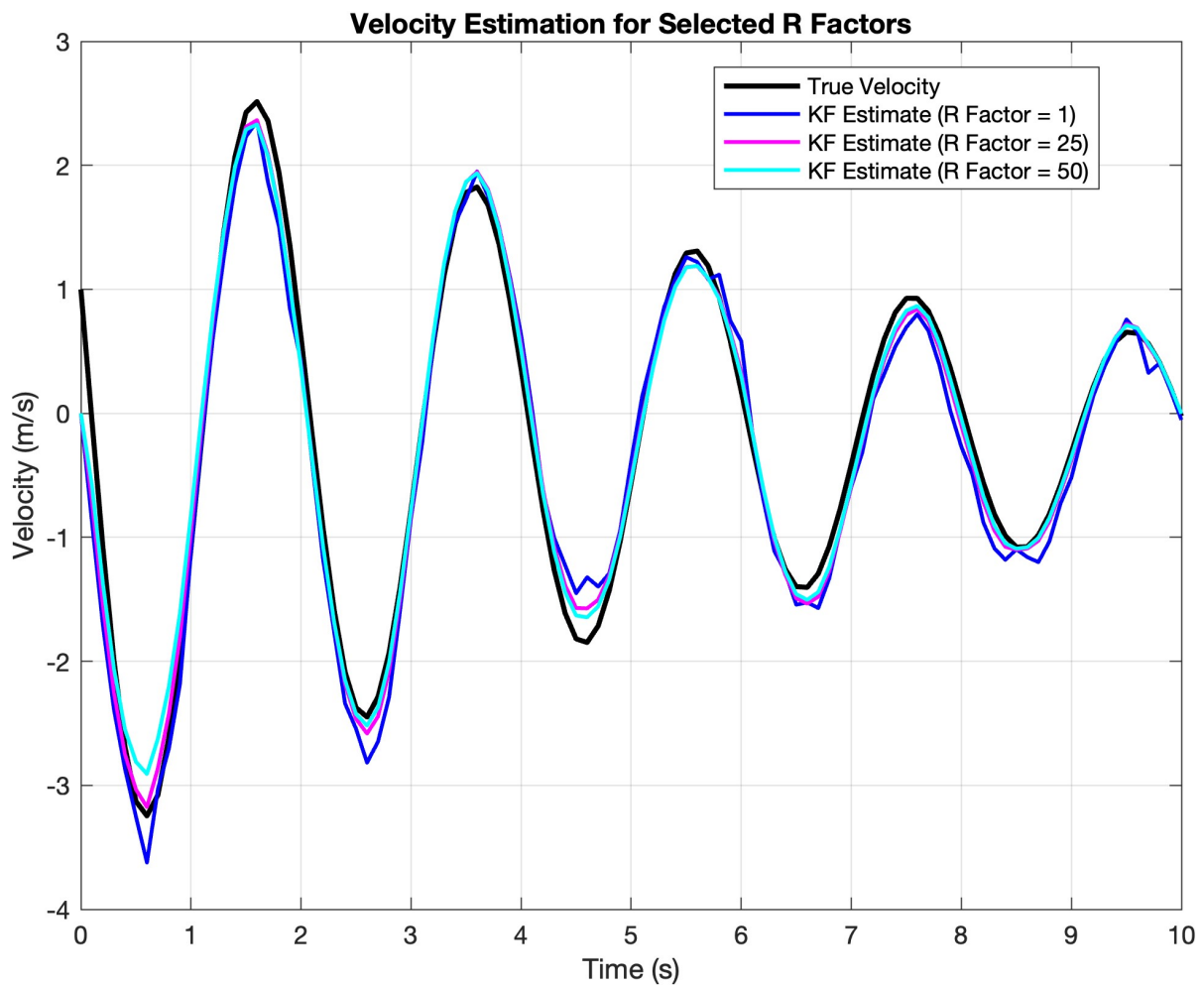


Figure 5.11: Velocity Estimates for Mass Spring System for Three Different Factors Multiplying the Covariance R.

Chapter 6

Conclusions and Future Work

In Chapter 2, we introduced the fundamental concepts of optimal control and the primary techniques used in this field. Building on these tools we derived the minimum energy estimator (MEE) and provided a detailed introduction to the minimum energy cost. In Chapter 3, we also developed the conditions for stability and convergence of the MEE.

Upon deriving an estimator equation for the MEE, we recognize the difficulty of obtaining an explicit solution due to the involvement of an Hamilton-Jacobi-Bellman (HJB) equation and the curse of dimensionality. To address these challenges, Chapter 4 explored numerical methods designed to circumvent this problem.

Chapter 5 focused on simulations that compared the performance of the discrete Kalman filter with that of a continuous Kalman filter with a late discretization. Additionally, we conducted simulations examining the effects of fine-tuning the noise covariance on the accuracy of the Kalman filter estimate for a mass spring problem in discrete time.

The primary goal of this thesis was to provide a clear and comprehensive introduction to the MEE, laying the groundwork for future research on related topics. This includes investigating and applying the numerical methods in Chapter 4 including Krener's MATLAB toolbox [57] for a classical approach or any of the machine learning and physics-informed neural network approaches detailed in the chapter. In terms of theory, there is work to be done relating to the relationship between early and late discretization for the MEE. Future work could also include finding a continuous, twice-differentiable, classical solution to the HJB, which would avoid having to invert $\frac{\partial^2}{\partial x^2} V(\hat{x}(t), t)$ in (3.24), the estimator equation for the continuous time MEE.

References

- [1] Murad Abu-Khalaf and Frank L. Lewis. Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach. *Automatica: The Journal of IFAC, the International Federation of Automatic Control*, 41(5):779–791, May 2005.
- [2] Marianne Akian, Ravindra Bapat, and Stéphane Gaubert. Max-plus algebra. In *Handbook of Linear Algebra*, pages 25–1–25–17. Chapman and Hall/CRC, 1st edition edition, 2 November 2006.
- [3] Alessandro Alla, Maurizio Falcone, and Luca Saluzzi. An efficient DP algorithm on a tree-structure for finite horizon optimal control problems. *SIAM Journal on Scientific Computing*, 25 July 2019.
- [4] Alessandro Alla and Luca Saluzzi. A HJB-POD approach for the control of nonlinear PDEs on a tree structure. *Applied Numerical Mathematics: Transactions of IMACS*, 155:192–207, 1 September 2020.
- [5] Alessandro Alla, Andreas Schmidt, and Bernard Haasdonk. Model order reduction approaches for infinite horizon optimal control problems via the HJB equation. In *Model Reduction of Parametrized Systems, Modeling, simulation & applications*, pages 333–347. Springer International Publishing, Cham, 2017.
- [6] Fabio Ancona and Alberto Bressan. Patchy vector fields and asymptotic stabilization. *ESAIM. Control, Optimisation and Calculus of Variations*, 4:445–471, 1999.
- [7] Kendall Atkinson. *An introduction to numerical analysis*. John Wiley & Sons, Nashville, TN, 2 edition, 3 January 1989.
- [8] Francis Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research: JMLR*, 18(19):1–53, 2017.

- [9] John S. Baras, Alain Bensoussan, and M. R. James. Dynamic observers as asymptotic limits of recursive filters: Special cases. *SIAM Journal on Applied Mathematics*, 48(5):1147–1158, 1988.
- [10] Andrew R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, May 1993.
- [11] Randal W. Beard, George N. Saridis, and John T. Wen. Approximate solutions to the time-invariant Hamilton-Jacobi-Bellman equation. *Journal of Optimization Theory and Applications*, 96(3):589–626, March 1998.
- [12] Richard E. Bellman. *Dynamic programming*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, Princeton, NJ, 1957.
- [13] Alain Bensoussan. *Stochastic Control of Partially Observable Systems*. Cambridge University Press, August 1992.
- [14] Alain Bensoussan, Jiayue Han, Sheung Chi Phillip Yam, and Xiang Zhou. Value-gradient based formulation of optimal control problem and machine learning algorithm. *SIAM Journal on Numerical Analysis*, 61(2):973–994, 30 April 2023.
- [15] Gal Berkooz, Philip Holmes, and John L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1):539–575, 1 January 1993.
- [16] Dimitri Bertsekas. *Reinforcement learning and Optimal Control*. Athena Scientific, 1 July 2019.
- [17] Gregory Beylkin and Martin J. Mohlenkamp. Numerical operator calculus in higher dimensions. *Proceedings of the National Academy of Sciences of the United States of America*, 99(16):10246–10251, 6 August 2002.
- [18] Olivier Bokanowski, Jochen Garcke, Michael Griebel, and Irene Klompaker. An adaptive sparse grid semi-Lagrangian scheme for first order Hamilton-Jacobi-Bellman equations. *Journal of Scientific Computing*, 55(3):575–605, 1 June 2013.
- [19] Tobias Breiten and Karl Kunisch. Neural network based nonlinear observers. *Systems & Control Letters*, 148:104829, 1 February 2021.
- [20] Simone Cacace, Emiliano Cristiani, Maurizio Falcone, and Athena Picarelli. A patchy dynamic programming scheme for a class of Hamilton-Jacobi-Bellman equations. *SIAM Journal on Scientific Computing*, 34(5):A2625–A2649, January 2012.
- [21] Fabio Camilli, Raul De Maio, and Elisa Iacomini. A hopf-lax formula for Hamilton-Jacobi equations with Caputo time-fractional derivative. *Journal of Mathematical Analysis and Applications*, 477(2):1019–1032, 15 September 2019.

- [22] Fabio Camilli and Espen R. Jakobsen. A finite element like scheme for integro-partial differential Hamilton-Jacobi-Bellman equations. *SIAM Journal on Numerical Analysis*, 47(4):2407–2431, 1 July 2009.
- [23] Dominique Chapelle, Asven Gariah, Philippe Moireau, and Jacques Sainte-Marie. A galerkin strategy with proper orthogonal decomposition for parameter-dependent problems – analysis, assessments and applications to parameter estimation. *ESAIM. Modelisation Mathématique et Analyse Numérique [ESAIM. Mathematical Modelling and Numerical Analysis]*, 47(6):1821–1843, November 2013.
- [24] Tao Cheng, Frank L. Lewis, and Murad Abu-Khalaf. Fixed-final-time-constrained optimal control of nonlinear systems using neural network HJB approach. *IEEE Transactions on Neural Networks*, 18(6):1725–1737, November 2007.
- [25] Yat Tin Chow, Jérôme Darbon, Stanley Osher, and Wotao Yin. Algorithm for overcoming the curse of dimensionality for state-dependent Hamilton-Jacobi equations. *Journal of Computational Physics*, 387:376–409, 15 June 2019.
- [26] Michael G. Crandall, Lawrence C. Evans, and Pierre-Louis Lions. Some properties of viscosity solutions of Hamilton-Jacobi equations. *Transactions of the American Mathematical Society*, 282(2):487–502, 1984.
- [27] Michael G. Crandall and Pierre-Louis Lions. Viscosity solutions of Hamilton-Jacobi equations. *Transactions of the American Mathematical Society*, 277(1):1–42, 1983.
- [28] Jérôme Darbon and Tingwei Meng. On some neural network architectures that can represent viscosity solutions of certain high dimensional Hamilton-Jacobi partial differential equations. *Journal of Computational Physics*, 425:109907, 15 January 2021.
- [29] Jérôme Darbon and Stanley Osher. Algorithms for overcoming the curse of dimensionality for certain Hamilton-Jacobi equations arising in control theory and elsewhere. *Research in the Mathematical Sciences*, 3(1):1–26, 1 December 2016.
- [30] Sergey Dolgov, Dante Kalise, and Karl K. Kunisch. Tensor decomposition methods for high-dimensional Hamilton–Jacobi–Bellman equations. *SIAM Journal on Scientific Computing: A Publication of the Society for Industrial and Applied Mathematics*, 43(3):A1625–A1650, 10 January 2021.
- [31] Sergey Dolgov, Dante Kalise, and Luca Saluzzi. Data-driven tensor train gradient cross approximation for Hamilton-Jacobi-Bellman equations. *SIAM Journal on Scientific Computing: A Publication of the Society for Industrial and Applied Mathematics*, 45(5):A2153–A2184, 2023.

- [32] J. R. Dormand and P. J. Prince. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1 March 1980.
- [33] Peter M. Dower, William M. McEneaney, and Huan Zhang. Max-plus fundamental solution semigroups for optimal control problems. In *2015 Proceedings of the Conference on Control and its Applications*, pages 368–375. Society for Industrial and Applied Mathematics, Philadelphia, PA, July 2015.
- [34] Weinan E, Jiequn Han, and Arnulf Jentzen. Algorithms for solving high dimensional PDEs: from nonlinear Monte Carlo to machine learning. *Nonlinearity*, 35(1):278–310, 6 January 2022.
- [35] Lawrence C. Evans. On solving certain nonlinear partial differential equations by accretive operator methods. *Israel Journal of Mathematics*, 36(3-4):225–247, September 1980.
- [36] Lawrence C. Evans. *Partial Differential Equations*. Graduate Studies in Mathematics. American Mathematical Society, Providence, RI, 15 May 1998.
- [37] Lawrence C. Evans. *An Introduction to Mathematical Optimal Control Theory*. 2021.
- [38] Maurizio Falcone and Roberto Ferretti. Semi-Lagrangian approximation schemes for linear and Hamilton-Jacobi equations. page 319, 2014.
- [39] Wendell H. Fleming and William M. McEneaney. A max-plus-based algorithm for a Hamilton-Jacobi-Bellman equation of nonlinear filtering. *SIAM Journal on Control and Optimization*, 38(3):683–710, 16 March 2000.
- [40] Wendell H. Fleming and Halil Mete Soner. *Controlled Markov processes and viscosity solutions*. Stochastic Modelling and Applied Probability. Springer, New York, NY, 2 edition, 17 November 2005.
- [41] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 42 of *Springer series in computational mathematics*. Springer, Berlin, Germany, 2012 edition, 24 February 2012.
- [42] Reinhard Hermann and Arthur J. Krener. Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, 22:728–740, 1977.
- [43] Omar Hijab. Minimum energy estimation. *PhD Dissertation*, 1980.
- [44] Eberhard Hopf. Generalized solutions of non-linear equations of first order. *Journal of Mathematics and Mechanics*, 14(6):951–973, 1965.
- [45] Matanya B. Horowitz, Anil Damle, and Joel W. Burdick. Linear Hamilton-Jacobi-Bellman equations in high dimensions. In *53rd IEEE Conference on Decision and Control*, pages 5880–5887, 2014.

- [46] Thomas Hunt and Arthur J. Krener. Improved patchy solution to the Hamilton-Jacobi-Bellman equations. pages 5835–5839, 15 December 2010.
- [47] Dario Izzo, Ekin Öztürk, and Marcus Mörtens. Interplanetary transfers via deep representations of the optimal policy and/or of the value function. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, New York, NY, USA, 13 July 2019. ACM.
- [48] Max Jensen and Iain Smears. On the convergence of finite element methods for Hamilton-Jacobi-Bellman equations. *SIAM Journal on Numerical Analysis*, 51(1):137–162, January 2013.
- [49] Frank Jiang, Glen Chou, Mo Chen, and Claire J. Tomlin. Using neural networks to compute approximate and guaranteed feasible Hamilton-Jacobi-Bellman PDE solutions. *arXiv [cs.LG]*, 9 November 2016.
- [50] Dante Kalise and Karl Kunisch. Polynomial approximation of high-dimensional Hamilton-Jacobi-Bellman equations and applications to feedback control of semi-linear parabolic PDEs. *SIAM Journal on Scientific Computing: A Publication of the Society for Industrial and Applied Mathematics*, 40(2):A629–A652, January 2018.
- [51] Rudolf E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1 March 1960.
- [52] Rudolf E. Kalman. On the general theory of control systems. *IFAC Proceedings Volumes*, 1(1):491–502, 1 August 1960.
- [53] Rudolf E. Kalman. Mathematical description of linear dynamical systems. *Journal of the Society for Industrial and Applied Mathematics*, 1(2):152–192, January 1963.
- [54] Wei Kang, Arthur J. Krener, Mingqing Xiao, and Liang Xu. A survey of observers for nonlinear dynamical systems. In Seon Ki Park and Liang Xu, editors, *Data Assimilation for Atmospheric, Oceanic and Hydrologic Applications (Vol. II)*, pages 1–25. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [55] Donald E. Kirk. *Optimal Control Theory: An Introduction*. Dover Books on Electrical Engineering Series. Dover Publications, Mineola, NY, 2004.
- [56] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.
- [57] Arthur J. Krener. Nonlinear systems toolbox.
- [58] Arthur J. Krener. The convergence of the minimum energy estimator. In Wei Kang, Carlos Borges, and Mingqing Xiao, editors, *New Trends in Nonlinear Dynamics and*

- Control and their Applications*, volume 295, pages 187–208. Springer Berlin Heidelberg, Berlin, Heidelberg, 1 May 2004.
- [59] Arthur J. Krener. Minimum energy estimation applied to the Lorenz attractor. In Maurizio Falcone, Roberto Ferretti, Lars Grüne, and William M McEneaney, editors, *Numerical Methods for Optimal Control Problems*, pages 165–182. Springer International Publishing, Cham, 2018.
 - [60] Arthur J. Krener and Kayo Ide. Measures of unobservability. In *Proceedings of the 48th IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 6401–6406. IEEE, December 2009.
 - [61] Karl Kunisch and Stefan Volkwein. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical Analysis*, 40(2):492–515, 1 February 2002.
 - [62] Karl Kunisch, Stefan Volkwein, and Le Xie. HJB-POD-based feedback design for the optimal control of evolution problems. *SIAM Journal on Applied Dynamical Systems*, 3(4):701–722, January 2004.
 - [63] Karl Kunisch and Daniel Walter. Semiglobal optimal feedback stabilization of autonomous systems via deep neural network approximation. *ESAIM. Control, Optimisation and Calculus of Variations*, 27:16, 2021.
 - [64] Harold J. Kushner. Numerical methods for stochastic control problems in continuous time. *SIAM Journal on Control and Optimization*, 28(5):999–1048, September 1990.
 - [65] Harold J. Kushner and Paul Dupuis. *Numerical methods for stochastic control problems in continuous time*. Stochastic Modelling and Applied Probability. Springer, New York, NY, 31 July 2012.
 - [66] Frank L Lewis, Lihua Xie, and Dan Popa. *Optimal and Robust Estimation: With an Introduction to Stochastic Control Theory, Second Edition*. Automation and Control Engineering. CRC Press, Boca Raton, FL, 2 edition, 17 September 2007.
 - [67] Daniel Liberzon. *Calculus of Variations and Optimal Control Theory: A Concise Introduction*. Princeton University Press, 8 January 2012.
 - [68] Tianyu Liu, Steven Ding, Jiarui Zhang, and Liutao Zhou. PINN-based viscosity solution of HJB equation. *arXiv [eess.SY]*, 18 September 2023.
 - [69] Hamed Masnadi-Shirazi, Alireza Masnadi-Shirazi, and Mohammad-Amir Dastgheib. A step by step mathematical derivation and tutorial on Kalman filters. *arXiv [stat.OT]*, 8 October 2019.

- [70] William M. McEneaney. *Max-plus methods for nonlinear control and estimation*. Systems & Control: Foundations & Applications. Birkhauser Boston, Secaucus, NJ, 2006 edition, 1 December 2005.
- [71] Leonard A. McGee and Stanley F. Schmidt. Discovery of the Kalman filter as a practical tool for aerospace and industry. Technical Report NASA-TM-86847, 1 November 1985.
- [72] Yiming Meng, Ruikun Zhou, Amartya Mukherjee, Maxwell Fitzsimmons, Christopher Song, and Jun Liu. Physics-informed neural network policy iteration: Algorithms, convergence, and verification. *arXiv [eess.SY]*, 15 February 2024.
- [73] Philippe Moireau. A discrete-time optimal filtering approach for nonlinear systems as a stable discretization of the Mortensen observer. *ESAIM. Control, Optimisation and Calculus of Variations*, 24(4):1815–1847, October 2018.
- [74] Kirsten A. Morris. *Course notes for AMath455/655 introduction to feedback control*. Department of Applied Mathematics University of Waterloo, 5 January 2015.
- [75] Richard E. Mortensen. Maximum-likelihood recursive nonlinear filtering. *Journal of Optimization Theory and Applications*, 2(6):386–394, 1 November 1968.
- [76] Saviz Mowlavi and Saleh Nabi. Optimal control of PDEs using physics-informed neural networks. *Journal of Computational Physics*, 473(111731):111731, January 2023.
- [77] Tenavi Nakamura-Zimmerer, Qi Gong, and Wei Kang. QRnet: Optimal regulator design with LQR-augmented neural networks. *IEEE Control Systems Letters*, 5:1303–1308, 2020.
- [78] Tenavi Nakamura-Zimmerer, Qi Gong, and Wei Kang. Adaptive deep learning for high-dimensional Hamilton-Jacobi-Bellman equations. *SIAM Journal on Scientific Computing: A Publication of the Society for Industrial and Applied Mathematics*, 43(2):A1221–A1247, January 2021.
- [79] Carmeliza Navasca and Arthur J. Krener. Patchy solutions of Hamilton-Jacobi-Bellman partial differential equations. In *Lecture Notes in Control and Information Sciences*, Lecture notes in control and information sciences, pages 251–270. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
- [80] Carmeliza Navasca and Arthur J. Krener. The patchy cost and feedback for the HJB PDE. *Proceedings of the 18th International Symposium on Mathematical Theory of Networks and Systems (Blacksburg VA, USA)*, 3 August 2008.

- [81] Ivan Oseledets. Tensor-train decomposition. *SIAM J. Scientific Computing*, 33:2295–2317, 2011.
- [82] Stanley Osher and Ronald Fedkiw. *Level set methods and dynamic implicit surfaces*. Applied mathematical sciences. Springer, New York, NY, 2003 edition, December 2003.
- [83] Stanley Osher and James A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1 November 1988.
- [84] Hans Josef Pesch and Michael Plail. The cold war and the maximum principle of optimal control. In Martin Grötschel, editor, *Optimization Stories*, pages 331–343. EMS Press, 1 edition, 1 January 2012.
- [85] Christophe Prud’homme, Dimitrios V. Rovas, Karen Veroy-Grepl, Lieven Machiels, Yvon Maday, Anthony T. Patera, and Gabriel Turinici. Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods. *Journal of Fluids Engineering*, 124(1):70–80, 1 March 2002.
- [86] Jean-Pierre Quadrat. *Min-plus linearity and statistical mechanics*. 1996.
- [87] Benjamin Recht. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):253–279, 3 May 2019.
- [88] Konrad Reif and Rolf Unbehauen. The extended Kalman filter as an exponential observer for nonlinear systems. *IEEE Transactions on Signal Processing: a Publication of the IEEE Signal Processing Society*, 47(8):2324–2328, 1999.
- [89] Lorenz Richter, Leon Sallandt, and Nikolas Nüsken. Solving high-dimensional parabolic PDEs using the tensor train format. *arXiv [stat.ML]*, 23 February 2021.
- [90] Leon Jasper Sallandt. Computing high-dimensional value functions of optimal feedback control problems using the tensor-train format, 2022.
- [91] Luca Saluzzi, Alessandro Alla, and Maurizio Falcone. Error estimates for a tree structure algorithm solving finite horizon control problems. *ESAIM. Control, Optimisation and Calculus of Variations*, 28:69, 2018.
- [92] James A. Sethian. *Cambridge monographs on applied and computational mathematics: Level set methods and fast marching methods: Evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science series number 3*. Cambridge monographs on applied and computational mathematics. Cambridge University Press, Cambridge, England, 2 edition, 13 June 1999.

- [93] Lawrence F. Shampine and Mark W. Reichelt. The MATLAB ODE suite. *SIAM Journal on Scientific Computing: A Publication of the Society for Industrial and Applied Mathematics*, 18(1):1–22, January 1997.
- [94] Alex Shum. Optimal direction-dependent path planning for autonomous vehicles. *PhD Thesis in Applied Mathematics*, 24 April 2014.
- [95] Dan Simon. *Optimal state estimation: Kalman, H-∞ and nonlinear approaches*. Wiley-Interscience, Hoboken, N.J., 2006.
- [96] Justin Sirignano and Konstantinos Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, December 2018.
- [97] Andrew Staniforth and Jean Côté. Semi-Lagrangian integration schemes for atmospheric models—a review. *Monthly Weather Review*, 119(9):2206–2223, 1 September 1991.
- [98] Yuval Tassa and Tom Erez. Least squares solutions of the HJB equation with neural network value-function approximators. *IEEE Transactions on Neural Networks*, 18(4):1031–1041, July 2007.
- [99] Emanuel Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences of the United States of America*, 106(28):11478–11483, 14 July 2009.
- [100] Hung Tran. *Hamilton-Jacobi Equations*, volume 213 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, 16 August 2021.
- [101] Kyriakos G. Vamvoudakis, Frank L. Lewis, and Shuzhi Sam Ge. *Neural networks in feedback control systems*, pages 1–52. John Wiley & Sons, Inc., Hoboken, NJ, USA, 20 February 2015.
- [102] Stefan Volkwein. Model reduction using proper orthogonal decomposition. *Lecture Notes, Institute of Mathematics and Scientific Computing, University of Graz*. see <http://www.uni-graz.at/imawww/volkwein/POD.pdf>, 1025, 2011.
- [103] Ivan Yegorov and Peter M. Dower. Characteristics in optimal control computation. In *Encyclopedia of Systems and Control*, pages 1–10. Springer London, London, 2020.
- [104] Ivan Yegorov and Peter M. Dower. Perspectives on characteristics based curse-of-dimensionality-free numerical approaches for solving Hamilton-Jacobi equations. *Applied Mathematics and Optimization*, 83(1):1–49, February 2021.

APPENDICES

Appendix A

Cost function derivation

For the derivation of the cost functional first introduced by Mortensen in [75] and named the maximum likelihood cost functional, we will start from:

$$\begin{cases} x_{t+1} = f(x_t) + gv_t \\ y_t = h(x_t) + kw_t \\ x_0 = x_\diamond + \chi_0. \end{cases} \quad (\text{A.1})$$

but in this section, $v(t) \in \mathbb{R}^n$ and $w(t) \in \mathbb{R}^m$ are assumed to be white noises rather than arbitrary disturbances:

$$\mathbb{E}[v(t)] = 0, \quad E[v(t) \cdot v^\top(\tau)] = Q\delta(t - \tau)$$

and

$$\mathbb{E}[w(t)] = 0, \quad E[w(t) \cdot w^\top(\tau)] = R\delta(t - \tau).$$

Here we are also letting the covariance matrices Q and R vary with time. Recall once more, the noise covariance matrix P_0 is positive semi-definite and Q_t, R_t are positive definite for all t . Note as well that covariance matrices are always symmetric.

This derivation is a detailed and extended version of Avneet Kaur's derivation.

Let $\{z_1, z_2, \dots, z_n\}$ be a set of random variables such that $z_i \sim \mathcal{N}(\mu, \sigma) \forall i \in \{1, 2, \dots, n\}$. Meaning, that the random variables are normally distributed with mean μ and variance σ .

Let's start with the likelihood function:

$$L_n(\mu, \sigma; \mathbf{z}) = \prod_{i=1}^k \mathbf{p}(\mathbf{z})$$

$$\begin{aligned}
L(\cdot) &= p(y_k, y_{k-1}, \dots, y_0, x_{k+1}, x_k, \dots, x_1, x_0) \\
L(\cdot) &= p(y_k, x_{k+1}, y_{k-1}, x_k, \dots, y_0, x_1, x_0) && \text{(re-arrange)} \\
L(\cdot) &= p(y_k, x_{k+1} | y_{k-1}, x_k, \dots) \cdot p(y_{k-1}, x_k, \dots) && \text{(use Bayes' Law)} \\
L(\cdot) &= p(y_k, x_{k+1} | x_k) \cdot p(y_{k-1}, x_k, \dots, y_0, x_1, x_0) && (x_{k+1} \text{ depends only on } x_k) \\
L(\cdot) &= p(y_k, x_{k+1} | x_k) \cdot p(y_{k-1} | x_k, \dots, y_0, x_1, x_0) && \text{(Bayes' Law for the second term)} \\
L(\cdot) &= p(x_0) \cdot \prod_{i=1}^k p(y_i, x_{i+1} | x_i) && \text{(generalize)} \\
L(\cdot) &= p(x_0) \cdot \prod_{i=1}^k \frac{p(y_i, x_{i+1}, x_i)}{p(x_i)} && \text{(Bayes' Law in reverse)} \\
L(\cdot) &= p(x_0) \cdot \prod_{i=1}^k \frac{p(y_i | x_{i+1}, x_i) p(x_{i+1}, x_i)}{p(x_i)} && \text{(Bayes' Law)} \\
L(\cdot) &= p(x_0) \cdot \prod_{i=1}^k \frac{p(y_i | x_{i+1}, x_i) p(x_{i+1} | x_i) p(x_i)}{p(x_i)} && \text{(Bayes' Law on second term)} \\
L(\cdot) &= p(x_0) \cdot \prod_{i=1}^k p(y_i | x_{i+1}, x_i) p(x_{i+1} | x_i) && \text{(simplify)} \\
L(\cdot) &= p(x_0) \cdot \prod_{i=1}^k p(y_i | x_i) p(x_{i+1} | x_i) && (y_i \text{ doesn't depend on future } x_{i+1})
\end{aligned}$$

with:

$$p(x_0) = \frac{1}{\sqrt{2\pi|P_0|}} \exp\left(\frac{1}{2}(x_0 - x_\diamond)^\top P_0^{-1}(x_0 - x_\diamond)\right)$$

so we obtain:

$$\begin{aligned}
L(\cdot) = p(x_0) &= \frac{1}{\sqrt{2\pi|P_0|}} \exp\left(-\frac{1}{2}(x_0 - x_\diamond)^\top P_0^{-1}(x_0 - x_\diamond)\right) \times \\
&\prod_{i=1}^k \left\{ \frac{1}{(2\pi)^{\frac{n}{2}} \sqrt{|Q|}} \exp\left(-\frac{1}{2}(x_{i+1} - f(x_i))^\top Q_i^{-1}(x_{i+1} - f(x_i))\right) \times \right. \\
&\quad \left. \frac{1}{(2\pi)^{\frac{n}{2}} \sqrt{|R|}} \exp\left(-\frac{1}{2}(y_i - h(x_i))^\top R_i^{-1}(y_i - h(x_i))\right) \right\}
\end{aligned}$$

now take log on both sides:

$$\begin{aligned} \log(L(\cdot)) = & \log\left(\frac{1}{\sqrt{2\pi|P_0|}}\right) - \left(\frac{1}{2}(x_0 - x_\diamond)^\top P_0^{-1}(x_0 - x_\diamond)\right) + \\ & \sum_{i=1}^k \left\{ \log\left(\frac{1}{(2\pi)^{\frac{n}{2}}\sqrt{|Q|}}\right) - \left(\frac{1}{2}(x_{i+1} - f(x_i))^\top Q_i^{-1}(x_{i+1} - f(x_i))\right) + \right. \\ & \left. \log\left(\frac{1}{(2\pi)^{\frac{n}{2}}\sqrt{|R|}}\right) - \left(\frac{1}{2}(y_i - h(x_i))^\top R_i^{-1}(y_i - h(x_i))\right) \right\} \end{aligned} \quad (\text{A.2})$$

Recall we are attempting to maximize the likelihood, so we want to maximize the log likelihood. Notice that maximizing (A.2) is the same as minimizing

$$\begin{aligned} J(\cdot) = & \frac{1}{2}(x_0 - x_\diamond)^\top P_0^{-1}(x_0 - x_\diamond) + \\ & \frac{1}{2} \sum_{i=1}^k \left\{ (x_{i+1} - f(x_i))^\top Q_i^{-1}(x_{i+1} - f(x_i)) + \right. \\ & \left. (y_i - h(x_i))^\top R_i^{-1}(y_i - h(x_i)) \right\}. \end{aligned} \quad (\text{A.3})$$

Rewritten using the weighted norms, we obtain:

$$J(T; \chi_0, v, w) = \frac{1}{2}\|x_0 - x_\diamond\|_{P_0}^2 + \frac{1}{2} \sum_{s=0}^t \left(\|x_{s+1} - f(x_s)\|_{Q^{-1}}^2 + \|y_s - h(x_s)\|_{R^{-1}}^2 \right).$$

We thus derived (3.3) starting from a stochastic framework and using the likelihood function.

Appendix B

Link to Code Repository

<https://github.com/albert-tres/thesiscode>